ON THE EXTENSIONS OF THE PREDICTOR-CORRECTOR PROXIMAL MULTIPLIER (PCPM) ALGORITHM AND THEIR APPLICATIONS

by

Run Chen

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

School of Industrial Engineering

West Lafayette, Indiana

December 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Andrdew L. Liu, Chair

School of Industrial Engineering

Dr. Gesualdo Scutari

School of Industrial Engineering

Dr. Susan R. Hunter

School of Industrial Engineering

Dr. Raghu Pasupathy

Department of Statistics

Approved by:

Dr. Abhijit Deshmukh

ACKNOWLEDGMENTS

First, I would like to thank my advisor Professor Andrew Liu. When I transferred my major from Physics to Operations Research, becoming a graduate student at School of Industrial Engineering, I knew nothing about optimization or power system. I have walked a long way until I finally graduate with doctoral degree, and my advisor is always by my side, supporting and guiding me along the way. From him, I have learnt not only the knowledge but also the way of doing research. His experience and insights are always of tremendous help when I get stuck at some problems. I really appreciate his great patience, especially for a student like me, who does not follow the advice every time. All my write-ups with his detailed comments, from a cover letter to a paper draft, are the things I will treasure for my whole life. In my eyes, Andrew Liu is a professor who really cares about the education and the development of a student. I feel extremely fortunate to have him as my advisor.

I would like to thank Professor Gesualdo Scutari, Professor Susan Hunter and Professor Raghu Pasupathy for agreeing to serve on my dissertation committee. I really appreciate them for all the valuable comments and suggestions.

I would also like to thank all the staff in my department for providing such a nice studying and working environment for us. I would say ”thank you” to all my colleagues on the third floor of Grissom Hall. Though I have not seen them for a long time due to the pandemic, their smiling faces have always brightened my day and will continue shine in my memory.

Finally, I would like to dedicate this dissertation to my parents. For nine years of studying abroad, I have missed too many important moments when they needed me. Without their unconditional love and support, I can never make a single step.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Many real-world application problems can be modeled mathematically as constrained convex optimization problems. The scale of such problems can be extremely large, posing significant challenges to traditional centralized algorithms and calling for efficient and scalable distributed algorithms. However, most of the existing works on distributed optimization have been focusing on block-separable problems with simple, linear constraints, such as the consensus-type constraints. The focus of this dissertation is to propose distributed algorithms to solve (possibly non-separable) large-scale optimization problems with more complicated constraints with parallel updating (aka in Jacobi fashion), instead of sequential updating in the form of Gauss-Seidel iterations. In achieving so, this dissertation extends the predictor corrector proximal multiplier method (PCPM) to address three issues when solving a large-scale constrained convex optimization problem: (i) non-linear coupling constraints; (ii) asynchronous iterative scheme; (iii) non-separable objective function and coupling constraints.

The idea of the PCPM algorithm is to introduce a predictor variable for the Lagrangian multiplier to avoid the augmented term, hence removing the coupling of block variables while still achieving convergence without restrictive assumptions. Building upon this algorithmic idea, we propose three distributed algorithms: (i) N-block PCPM algorithm for solving N-block convex optimization problems with both linear and nonlinear coupling constraints; (ii) asynchronous N-block PCPM algorithm for solving linearly constrained N-block convex optimization problems; (iii) a distributed algorithm, named PC^{2}PM, for solving large-scale convex quadratically constrained quadratic programs (QCQPs). The global convergence is established for each of the three algorithms. Extensive numerical experiments, using various data sets, are conducted on either a single-node computer or a multi-node computer cluster through message passing interface (MPI). Numerical results demonstrate the efficiency and scalability of the proposed algorithms.

A real application of the N-block PCPM algorithm to solve electricity capacity expansion models is also studied in this dissertation. A hybrid scenario-node-realization decomposition method, with extended nonanticipativity constraints, is proposed for solving the resulting

large-scale optimization problem from a multi-scale, multi-stage stochastic program under various uncertainties with different temporal scales. A technique of orthogonal projection is exploited to simplify the iteration steps, which leads to a simplified N-block PCPM algorithm amenable to massive parallelization during iterations. Such an algorithm exhibits much more scalable numerical performance when compared with the widely used progressive hedging approach (PHA) for solving stochastic programming problems.

1. INTRODUCTION

1.1 Motivation

Many real-world application problems can be modeled mathematically as constrained convex optimization problems, as follows:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X}, \\ & A\mathbf{x} = \mathbf{b}, \\ & g_{\mathrm{j}}(\mathbf{x}) \leq 0, \quad \mathrm{j} = 1, \ldots, M, \end{aligned} \tag{1.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the decision variable, the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function, and each nonlinear constraint function $g_{\mathrm{j}} : \mathbb{R}^n \to \mathbb{R}$ is a convex function for all $\mathrm{j} = 1, \ldots, M$. The constraint set $\mathcal{X} \subset \mathbb{R}^n$ is a closed convex set, $A \in \mathbb{R}^{m \times n}$ is a full row-rank matrix, and $\mathbf{b} \in \mathbb{R}^m$ is a given vector.

However, the scale of the optimization problem (1.1), characterized by n, m and M, can be extremely large due to many factors, such as the availability of large-scale data and the increasing complexity of physical/cyber systems. Such problems pose significant challenges to traditional centralized algorithms, such as the well-established interior point method (IPM) and the sequential quadratic programming (SQP) method [1], which cannot be easily carried out on multiple computing units to speed up convergence to an optimal solution. Moreover, the exploding volume of data (analytical form of functions, matrices and vectors) may overwhelm the storage of a single computing unit. Even when there are no out-of-memory issues, it will take a really long time for the centralized algorithm to converge, even with polynomial running time.

To overcome these limitations of centralized algorithms, various decomposition techniques have been developed [2]. For example, the alternating direction method of multipliers (ADMM) has been widely used to solve large-scale optimization problems arising from statistics and machine learning [3]. To illustrate our point and ease the argument, we briefly describe the ADMM algorithm below, as well as the PCPM algorithm, which is another decomposition approach.

- **Alternating Direction Method of Multipliers (ADMM)**

Consider the following 2-block linearly constrained convex optimization problem:

$$\begin{aligned} \underset{\mathbf{x}_1 \in \mathbb{R}^{n_1},\ \mathbf{x}_2 \in \mathbb{R}^{n_2}}{\text{minimize}} \quad & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) \\ \text{subject to} \quad & A_1\mathbf{x}_1 + A_2\mathbf{x}_2 = \mathbf{b}, \quad (\boldsymbol{\lambda}) \end{aligned} \tag{1.2}$$

where $f_1 : \mathbb{R}^{n_1} \to (-\infty, +\infty]$ and $f_2 : \mathbb{R}^{n_2} \to (-\infty, +\infty]$ are closed proper convex functions, $A_1 \in \mathbb{R}^{m\times n_1}$ and $A_2 \in \mathbb{R}^{m\times n_2}$ are full row-rank matrices, $\mathbf{b} \in \mathbb{R}^m$ is a given vector, and $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the corresponding Lagrangian multiplier associated with the linear equality constraint. The classic Lagrangian function $\mathcal{L} : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^m \to \mathbb{R}$ is defined as:

$$\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda}) = f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + \boldsymbol{\lambda}^T(A_1\mathbf{x}_1 + A_2\mathbf{x}_2 - \mathbf{b}). \tag{1.3}$$

It is well-known that for a convex problem of the specific form in (1.2) (where the linear constraint qualification automatically holds), finding an optimal solution is equivalent to finding a saddle point $(\mathbf{x}_1^*, \mathbf{x}_2^*, \boldsymbol{\lambda}^*)$ such that $\mathcal{L}(\mathbf{x}_1^*, \mathbf{x}_2^*, \boldsymbol{\lambda}) \le \mathcal{L}(\mathbf{x}_1^*, \mathbf{x}_2^*, \boldsymbol{\lambda}^*) \le \mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda}^*)$ [4]. To find such a saddle point, a simple dual decomposition algorithm can be applied to $\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda})$. More specifically, at each iteration k, given a fixed Lagrangian multiplier $\boldsymbol{\lambda}^k$, the primal decision variables $(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1})$ can be obtained, in parallel, by minimizing $\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda}^k)$. Then a dual update $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2^{k+1} - \mathbf{b})$ is performed, where a positive scalar ρ is the step size.

While the above algorithmic idea is simple, it is well-known that convergence cannot be established without more restrictive assumptions, such as strict convexity of f_1 and f_2 (e.g., Theorem 26.3 in [4]). One approach to overcome such difficulties is to obtain the primal decision variables $(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1})$ via minimizing the augmented Lagrangian function, evaluated at $\boldsymbol{\lambda}^k$: $\mathcal{L}_\rho(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda}^k) := \mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \boldsymbol{\lambda}^k) + \frac{\rho}{2}\|A_1\mathbf{x}_1 + A_2\mathbf{x}_2 - \mathbf{b}\|_2^2$, in an alternating direction:

$$\begin{aligned} \mathbf{x}_1^{k+1} =& \underset{\mathbf{x}_1 \in \mathbb{R}^{n_1}}{\text{argmin}} \quad f_1(\mathbf{x}_1) + (\boldsymbol{\lambda}^k)^T A_1\mathbf{x}_1 + \frac{\rho}{2}\|A_1\mathbf{x}_1 + A_2\mathbf{x}_2^k - \mathbf{b}\|_2^2, \\ \mathbf{x}_2^{k+1} =& \underset{\mathbf{x}_2 \in \mathbb{R}^{n_2}}{\text{argmin}} \quad f_2(\mathbf{x}_2) + (\boldsymbol{\lambda}^k)^T A_2\mathbf{x}_2 + \frac{\rho}{2}\|A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2 - \mathbf{b}\|^2, \end{aligned} \tag{1.4}$$

followed by the dual update:

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2^{k+1} - \mathbf{b}). \tag{1.5}$$

The parameter ρ is given, which determines the step-size for primal and dual variables' update in each iteration, and recent studies of convergence properties about ADMM can be found in [5], [6]. With (1.4), however, $\mathbf{x}_1^{k+1}$ and $\mathbf{x}_2^{k+1}$ can no longer be obtained in parallel due to the augmented term $\|A_1\mathbf{x}_1 + A_2\mathbf{x}_2 - \mathbf{b}\|_2^2$. To overcome this difficulty, the predictor corrector proximal multiplier method (PCPM) is proposed by [7].

• **Predictor Corrector Proximal Multiplier Method (PCPM)**

For the given Lagrangian multiplier $\boldsymbol{\lambda}^k$, the PCPM algorithm introduces a predictor variable $\boldsymbol{\mu}^{k+1}$:

$$\boldsymbol{\mu}^{k+1} := \boldsymbol{\lambda}^k + \rho(A_1\mathbf{x}_1^k + A_2\mathbf{x}_2^k - \mathbf{b}). \tag{1.6}$$

Using the predictor variable, the primal decision variables $(\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1})$ are obtained via minimizing the classic Lagrangian function, evaluated at $\boldsymbol{\mu}^{k+1}$, plus two proximal terms:

$$\begin{aligned} (\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}) = \underset{\mathbf{x}_1 \in \mathbb{R}^{n_1}, \mathbf{x}_2 \in \mathbb{R}^{n_2}}{\text{argmin}} & f_1(\mathbf{x}_1) + f_2(\mathbf{x}_2) + (\boldsymbol{\mu}^{k+1})^T(A_1\mathbf{x}_1 + A_2\mathbf{x}_2 - \mathbf{b}) \\ & + \frac{1}{2\rho}\|\mathbf{x}_1 - \mathbf{x}_1^k\|_2^2 + \frac{1}{2\rho}\|\mathbf{x}_2 - \mathbf{x}_2^k\|_2^2, \end{aligned} \tag{1.7}$$

which allows $\mathbf{x}_1^{k+1}$ and $\mathbf{x}_2^{k+1}$ to be updated in parallel again. After solving (1.7), the PCPM algorithm updates the dual variable as follows:

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(A_1\mathbf{x}_1^{k+1} + A_2\mathbf{x}_2^{k+1} - \mathbf{b}), \tag{1.8}$$

which is referred to as a corrector update.

The above two distributed algorithms, both of the type of proximal point algorithms, can be conveniently applied to solve 2-block linearly constrained convex optimization problems. However, there are still some issues for solving large-scale optimization problems, especially in the general form of (1.1), that can not be addressed:

(i) **nonlinear coupling constraints**

Let the constrained convex optimization problem (1.1) be of the following specific form:

$$\begin{aligned} \underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & f(\mathbf{x}_1,\ldots,\mathbf{x}_N) = \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) \\ \text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \mathrm{i}=1,\ldots,N, \\ & \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} = \mathbf{b}, \\ & g_{\mathrm{j}}(\mathbf{x}_1,\ldots,\mathbf{x}_N) = \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}) \le 0, \quad \mathrm{j}=1,\ldots,M, \end{aligned} \tag{1.9}$$

where each decision variable $\mathbf{x}_{\mathrm{i}} \in \mathbb{R}^{n_{\mathrm{i}}}$ is regarded as a *block* for all $\mathrm{i}=1,\ldots,N$, and $\sum_{\mathrm{i}=1}^{N} n_{\mathrm{i}} = n$. Accordingly, the objective function $f:\mathbb{R}^n \to \mathbb{R}$ can be expressed as a summation of N functions, where each function $f_{\mathrm{i}}:\mathbb{R}^{n_{\mathrm{i}}} \to \mathbb{R}$ is defined on the single block $\mathbf{x}_{\mathrm{i}}$ and is assumed to be convex for all $\mathrm{i}=1,\ldots,N$. The constraint set $\mathcal{X}$ can be expressed as a Cartesian product of N disjoint sets: $\mathcal{X} = \prod_{\mathrm{i}=1}^{N} \mathcal{X}_{\mathrm{i}}$, where each $\mathcal{X}_{\mathrm{i}} \subset \mathbb{R}^{n_{\mathrm{i}}}$ is a closed convex set for all $\mathrm{i}=1,\ldots,N$. The matrix A in the linear equality constraint can be expressed as $A = (A_1 \ldots A_N)$, where each $A_{\mathrm{i}} \in \mathbb{R}^{m\times n_{\mathrm{i}}}$ for all $\mathrm{i}=1,\ldots,N$. Each nonlinear constraint function $g_{\mathrm{j}}:\mathbb{R}^n \to \mathbb{R}$ can also be expressed as a summation of N functions, where each function $g_{\mathrm{ji}}:\mathbb{R}^{n_{\mathrm{i}}} \to \mathbb{R}$ is assumed to be convex for all $\mathrm{i}=1,\ldots,N$ and $\mathrm{j}=1,\ldots,M$. Such a problem is then called a *block-separable* convex optimization problem with both linear and nonlinear coupling constraints. Note that the N-block nonlinear coupling constraints can not be directly handled by ADMM-type or PCPM-type algorithms; thus the algorithmic ideas for decomposing such nonlinear coupling need to be explored.

(ii) **asynchronous iterative scheme**

While solving the decomposed sub-problems in parallel, the scale of each sub-problem may vary significantly, which likely leads to significant differences in computation time among all computing units. Moreover, different communication efficiency among all units contributes to differences in communication delay. For synchronous, distributed algorithms with an iterative scheme, the start of each iteration must be simultaneous for all computing units. The next iteration will not start until all units finish computation and communication. These issues can severely slow down the speed of distributed algorithms, as the time spent on each iteration is determined by the slowest unit. To overcome these limitations of synchronized distributed algorithms, developing convergent, asynchronous distributed algorithms is of great significance.

(iii) **non-separable objective function and coupling constraints**

Consider a large-scale constrained optimization problem (1.1) with non-separable objective function and nonlinear coupling constrains:

$$\begin{aligned} \underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & f(\mathbf{x}_1,\ldots,\mathbf{x}_N) \\ \text{subject to} \quad & \mathbf{x}_\mathrm{i} \in \mathcal{X}_\mathrm{i}, \quad \mathrm{i} = 1,\ldots,N, \\ & \sum_{\mathrm{i}=1}^{N} A_\mathrm{i}\mathbf{x}_\mathrm{i} = \mathbf{b}, \\ & g_\mathrm{j}(\mathbf{x}_1,\ldots,\mathbf{x}_N) \le 0, \quad \mathrm{j} = 1,\ldots,M. \end{aligned} \tag{1.10}$$

Developing efficient distributed algorithms to decompose such large-scale convex optimization problems with non-separable and nonlinear coupling constraints, while still achieving convergence under proper assumptions, is of great research interest as well as real application need [8], [9].

1.2 Objectives of the Thesis

The main objective of the thesis is to develop efficient and scalable distributed algorithms to address the three aforementioned issues. We propose three distributed algorithms and establish their global convergence; each algorithm addresses one of the three issues respectively.

We then apply the proposed distributed algorithms to solve large-scale constrained convex optimization problems arising from real-world application areas, including machine learning and power systems. Numerical experiments using both synthetic and real large-scale data sets are implemented on massive parallel distributed computing units to demonstrate real-world performance and for the purpose of comparison with other algorithms.

1.3 Thesis Outline

The remainder of the dissertation is organized as follows. In Chapter 2, we propose an extended N-block PCPM algorithm to solve N-block convex optimization problems with both linear and nonlinear coupling constrains. We establish the global convergence under mild assumptions, and further establish the algorithm's linear convergence rate under stronger conditions. Numerical experiments demonstrate its performance. Moreover, we also propose an asynchronous N-block PCPM algorithm with the standard bounded delay assumption, first for linearly constrained N-block convex optimization problems as a starting point. We establish the global sub-linear convergence rate with the additional assumption of strong convexity of the objective function. The proposed algorithm is applied to solve a graph optimization problem arising from spatial clustering. The numerical results demonstrate the convergence and the efficiency of the asynchronous iterative scheme.

In Chapter 3, we propose a Jacobi-style distributed algorithm to solve convex quadratically constrained quadratic programs (QCQPs), using a novel idea of predictor-corrector primal-dual update with an adaptive step size. The algorithm enables distributed storage of data as well as parallel distributed computing. We establish the conditions for the proposed algorithm to converge to a global optimum, and implement our algorithm on a computer cluster with multiple nodes using Message Passing Interface (MPI). The numerical experiments are conducted on data sets of various scales from different applications, and the results show that our algorithm exhibits favorable scalability for solving large-scale problems.

In Chapter 4, we apply the N-block PCPM algorithm to solve electricity capacity expansion models, formulated as a multi-scale, multi-stage stochastic program. Different from the traditional scenario decomposition method, we propose a hybrid scenario-node-realization decomposition method, with extended nonanticipativity constraints, that can decompose the large-scale problem under various uncertainties with different temporal scales. While applying the N-block PCPM algorithm to solve the resulting deterministic, large-scale N-block convex optimization problem, we exploit a technique of orthogonal projection, which greatly simplifies the iteration of the N-block PCPM algorithm and save the communication among all computing units. The proposed simplified N-block PCPM algorithm, along with the hybrid decomposition method, is applied to solve a capacity expansion model with both synthetic and historical data, and is implemented on a multi-node computer cluster using MPI. We compare the performance with the ADMM algorithm and the PHA algorithm, and the numerical results demonstrate the scalability of the proposed algorithm when solving the resulting large-scale block-separable optimization problems.

Chapter 5 summarizes the results of the dissertation and concludes with some discussions of interesting future research.

2. DISTRIBUTED AND ASYNCHRONOUS ALGORITHMS FOR N-BLOCK CONVEX OPTIMIZATION WITH COUPLING CONSTRAINTS

2.1 Introduction

In this work, we focus on designing a distributed algorithm for solving block-separable convex optimization problems with both linear and nonlinear coupling constraints, discussed in Section 1.1. More specifically, we consider the following problem:

$$\begin{aligned}
\underset{\mathbf{x}_1,\dots,\mathbf{x}_N}{\text{minimize}} \quad & f(\mathbf{x}_1,\dots,\mathbf{x}_N) = \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) \\
\text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \mathrm{i} = 1,\dots,N, \\
& \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} = \mathbf{b}, \\
& g_{\mathrm{j}}(\mathbf{x}_1,\dots,\mathbf{x}_N) = \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}) \le 0, \quad \mathrm{j} = 1,\dots,M,
\end{aligned} \tag{2.1}$$

where each block of decision variable $\mathbf{x}_{\mathrm{i}} \in \mathbb{R}^{n_{\mathrm{i}}}$ is constrained by a closed and convex set $\mathcal{X}_{\mathrm{i}} \subset \mathbb{R}^{n_{\mathrm{i}}}$ for all $\mathrm{i} = 1,\dots,N$, and $\sum_{\mathrm{i}=1}^{N} n_{\mathrm{i}} = n$. The objective function $f : \mathbb{R}^n \to \mathbb{R}$ is block-separable, and each function $f_{\mathrm{i}} : \mathbb{R}^{n_{\mathrm{i}}} \to \mathbb{R}$ is assumed to be continuous and convex for all $\mathrm{i} = 1,\dots,N$. All blocks $\mathbf{x}_{\mathrm{i}}$'s are coupled in a linear equality constraint, where each $A_{\mathrm{i}} \in \mathbb{R}^{m \times n_{\mathrm{i}}}$ is a given matrix for all $\mathrm{i} = 1,\dots,N$, and $\mathbf{b} \in \mathbb{R}^m$ is a given vector. All blocks $\mathbf{x}_{\mathrm{i}}$'s are also coupled in a system of nonlinear inequality constraints, where each constraint function $g_{\mathrm{j}} : \mathbb{R}^n \to \mathbb{R}$ is also block-separable and each function $g_{\mathrm{ji}} : \mathbb{R}^{n_{\mathrm{i}}} \to \mathbb{R}$ is assumed to be continuous and convex for all $\mathrm{i} = 1,\dots,N$ and $\mathrm{j} = 1,\dots,M$. A wide range of application problems can be mathematically formulated as optimization problems of the form (2.1), arising from the areas including optimal control [10], network optimization [11], statistical learning [3] and etc.

The alternating direction method of multipliers (ADMM) [3], as well as its variants [12]–[14], is an efficient distributed algorithm for solving convex block-separable optimization problems with linear coupling constraints, but problems of (2.1) with nonlinear coupling constraints can not be directly handled by ADMM-typed algorithms.

To overcome the above-mentioned limitations of the ADMM-typed algorithms, we first extend the 2-block PCPM algorithm to solve an N-block convex optimization problem with both linear and nonlinear coupling constraints. We further extend the N-block PCPM algorithm to an asynchronous iterative scheme, where a maximum tolerable delay is allowed for each distributed unit, and apply it to solve an N-block convex optimization problem with general linear coupling constraints.

The remainder of the chapter is organized as follows. In Section 2.2, we present an extended N-block PCPM algorithm for solving general constrained N-block convex optimization problems. We first establish global convergence under mild assumptions, and then prove the linear convergence rate with slightly stronger assumptions. In Section 2.3, we further extend the N-block PCPM algorithm to an asynchronous scheme with the bounded delay assumption. We establish both convergence and global sub-linear convergence rate under the conditions of strong convexity. Section 2.4 presents the numerical results of applying the proposed algorithms to solve a graph optimization problem arising from an application of housing price prediction. Finally, Section 2.5 concludes this chapter with discussions of the limitations of the algorithms and possible future research directions.

2.2 Extending the PCPM Algorithm to Solving General Constrained N-block Convex Optimization Problems

2.2.1 N-block PCPM Algorithm for General Constrained Convex Optimization Problems

In the 2-block PCPM algorithm presented in Section 1.1, we observe that the introduction of the predictor variable eliminates the quadratic term in the proximal augmented Lagrangian function and make the primal minimization step parallelizable again, which is a major difference from the ADMM algorithm. For an N-block convex optimization problem with additional nonlinear coupling constraints:

$$\begin{aligned}
\underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) \\
\text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \mathrm{i} = 1, \ldots, N, \\
& \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} = \mathbf{b}, \quad (\boldsymbol{\lambda}) \\
& \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}) \leq 0, \quad \mathrm{j} = 1, \ldots, M, \quad (\mu_{\mathrm{j}})
\end{aligned} \tag{2.2}$$

the potential coupling caused by the quadratic term $\|\sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}})\|_2^2$ could be even worse. Using the same technique of introducing the predictor variable, we extend the 2-block PCPM algorithm for solving (2.2).

First, we make a blanket assumption on problem (2.2) throughout this chapter that the Slater's constraint qualification (CQ) holds.

Assumption 2.2.1 (Slater's CQ)**.** There exists a point $(\bar{\mathbf{x}}_1, \ldots, \bar{\mathbf{x}}_N)$ such that

$$\left\{ \bar{\mathbf{x}}_{\mathrm{i}} \in relint(\mathcal{X}_{\mathrm{i}}), \quad \mathrm{i} = 1, \ldots, N \;\middle|\; \begin{array}{l} \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\bar{\mathbf{x}}_{\mathrm{i}} = \mathbf{b} \\ \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\bar{\mathbf{x}}_{\mathrm{i}}) < 0, \quad \mathrm{j} = 1, \ldots, M \end{array} \right\},$$

where $relint(\mathcal{X}_{\mathrm{i}})$ denotes the relative interior of the convex set $\mathcal{X}_{\mathrm{i}}$ for all $\mathrm{i} = 1, \ldots, N$.

To apply the PCPM algorithm to (2.2), at each iteration k, with a given primal dual pair, $\left(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k := (\mu_1^k, \ldots, \mu_M^k)^T\right)$, we start with a predictor update:

$$\begin{aligned}
& \textbf{(predictor update)} : \\
& \boldsymbol{\gamma}^{k+1} = \boldsymbol{\lambda}^k + \rho\Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^k - \mathbf{b}\Big), \\
& \nu_{\mathrm{j}}^{k+1} = \Pi_{\mathbb{R}_+}\Big[\mu_{\mathrm{j}}^k + \rho\sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}^k)\Big], \quad \mathrm{j} = 1, \ldots, M,
\end{aligned} \tag{2.3}$$

where $\Pi_{\mathbb{Z}}(\mathbf{z})$ denotes the projection of a vector $\mathbf{z} \in \mathbb{R}^n$ onto a set $\mathbb{Z} \subset \mathbb{R}^n$, and $\mathbb{R}_+$ refers to the set of all non-negative real numbers.

After the predictor update, we update the primal variables $(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1})$ by minimizing the Lagrangian function $\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})$ evaluated at the predictor variable $\left(\boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1} := (\nu_1^{k+1}, \ldots, \nu_M^{k+1})^T\right)$, plus the proximal terms. The primal minimization step can be decomposed as

$$\begin{aligned}&\textbf{(primal minimization)}:\\ &\mathbf{x}_{\mathrm{i}}^{k+1} = \operatorname*{argmin}_{\mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}} \; f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) + (\boldsymbol{\gamma}^{k+1})^T A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}} + \sum_{\mathrm{j}=1}^{M} \nu_{\mathrm{j}}^{k+1} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}) + \frac{1}{2\rho} \|\mathbf{x}_{\mathrm{i}} - \mathbf{x}_{\mathrm{i}}^k\|_2^2,\\ &\mathrm{i} = 1, \ldots, N.\end{aligned} \tag{2.4}$$

A corrector update is then performed for each Lagrangian multiplier $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$:

$$\begin{aligned}&\textbf{(dual corrector)}:\\ &\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho\Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{b}\Big),\\ &\mu_{\mathrm{j}}^{k+1} = \Pi_{\mathbb{R}_+}\Big[\mu_{\mathrm{j}}^k + \rho \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}^{k+1})\Big], \quad \mathrm{j} = 1, \ldots, M.\end{aligned} \tag{2.5}$$

The overall structure of N-block PCPM algorithm is presented in Algorithm 1 below.

Algorithm 1 N-PCPM

```
1: Initialization choose an arbitrary starting point $(\mathbf{x}_1^0, \ldots, \mathbf{x}_N^0, \boldsymbol{\lambda}^0, \boldsymbol{\nu}^0)$.
2: $k \leftarrow 0$.
3: while termination conditions are not met do
4:     (Predictor update)
        update $(\boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})$ according to (2.3);
5:     (Primal minimization)
        update $(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1})$ according to (2.4);
6:     (Corrector update)
        update $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$ according to (2.5);
7:     $k \leftarrow k + 1$
8: return $(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\nu}^k)$.
```

2.2.2 Convergence Analysis

We make the following additional assumptions on the optimization problem (2.2).

Assumption 2.2.2 (Lipschitz Continuity)**.** For all $\mathrm{j} = 1 \dots M$ and $\mathrm{i} = 1 \dots N$, each single-valued function $g_{\mathrm{ij}} : \mathcal{X}_{\mathrm{i}} \to \mathbb{R}$ is Lipschitz continuous with modulus of L_{ji}, i.e., $\|g_{\mathrm{ji}}(\mathbf{x}_1) - g_{\mathrm{ji}}(\mathbf{x}_2)\|_2 \le L_{\mathrm{ji}}\|\mathbf{x}_1 - \mathbf{x}_2\|_2$ for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}_{\mathrm{i}}$.

Assumption 2.2.3 (Existence of a Saddle Point)**.** For the Lagrangian function of (2.2):

$$\mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_N, \boldsymbol{\lambda}, \boldsymbol{\mu}) := \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) + \boldsymbol{\lambda}^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} - \mathbf{b} \Big) + \sum_{\mathrm{j}=1}^{M} \mu_{\mathrm{j}} \sum_{\mathrm{i}=1}^{N} g_{\mathrm{ji}}(\mathbf{x}_{\mathrm{i}}), \tag{2.6}$$

we assume that a saddle point $(\mathbf{x}_1^*, \dots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ exists; that is, for any $\mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}$, $\mathrm{i} = 1, \dots, N$, $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\boldsymbol{\mu} \in \mathbb{R}_+^M$,

$$\mathcal{L}(\mathbf{x}_1^*, \dots, \mathbf{x}_N^*, \boldsymbol{\lambda}, \boldsymbol{\mu}) \le \mathcal{L}(\mathbf{x}_1^*, \dots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \le \mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_N, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*). \tag{2.7}$$

Note that coupled with the blanket Assumption 2.2.1 that Slater's CQ holds for the optimization problem (2.2), the above assumption is equivalent to say that an optimal solution of (2.2) is assumed to exist (see Corollary 28.3.1 in [4]).

Next, we derive some essential lemmas for constructing the main convergence proof. The following lemma is due to Proposition 6 in [15], and for completeness, we provide the detailed statements below.

Lemma 2.2.1 (Inequality of Proximal Minimization Point)**.** Given a closed, convex set $\mathbb{Z} \subset \mathbb{R}^n$, and a continuous, convex function $F : \mathbb{Z} \to \mathbb{R}$. With a given point $\bar{\mathbf{z}} \in \mathbb{Z}$ and a positive number $\rho > 0$, if $\hat{\mathbf{z}}$ is a proximal minimization point; i.e. $\hat{\mathbf{z}} := \arg\min_{\mathbf{z} \in \mathbb{Z}} F(\mathbf{z}) + \frac{1}{2\rho}\|\mathbf{z} - \bar{\mathbf{z}}\|_2^2$, then we have that

$$2\rho[F(\hat{\mathbf{z}}) - F(\mathbf{z})] \le \|\bar{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\hat{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\hat{\mathbf{z}} - \bar{\mathbf{z}}\|_2^2, \quad \forall \mathbf{z} \in \mathbb{Z}. \tag{2.8}$$

Proof. Denote $\Phi(\mathbf{z}) = F(\mathbf{z}) + \frac{1}{2\rho}\|\mathbf{z} - \bar{\mathbf{z}}\|_2^2$. By the definition of $\hat{\mathbf{z}}$, we have $\mathbf{0} \in \partial_{\mathbf{z}}\Phi(\hat{\mathbf{z}})$. Since $\Phi(\mathbf{z})$ is strongly convex with modulus $\frac{1}{\rho}$, it follows that $2\rho\big[\Phi(\mathbf{z}) - \Phi(\hat{\mathbf{z}})\big] \ge \|\hat{\mathbf{z}} - \mathbf{z}\|_2^2$ for any $\mathbf{z} \in \mathbb{Z}$. □

Lemma 2.2.2. The update steps (2.3) and (2.5) are equivalent to obtaining proximal minimization points as follows:

$$\begin{aligned}(\boldsymbol{\gamma}^{k+1},\boldsymbol{\nu}^{k+1}) = \underset{\boldsymbol{\lambda}\in\mathbb{R}^m,\boldsymbol{\mu}\in\mathbb{R}^M_+}{\operatorname{argmin}} \quad & -\mathcal{L}(\mathbf{x}_1^k,\dots,\mathbf{x}_N^k,\boldsymbol{\lambda},\boldsymbol{\mu}) \\ & +\frac{1}{2\rho}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^k\|_2^2+\frac{1}{2\rho}\|\boldsymbol{\mu}-\boldsymbol{\mu}^k\|_2^2, \\ (\boldsymbol{\lambda}^{k+1},\boldsymbol{\mu}^{k+1}) = \underset{\boldsymbol{\lambda}\in\mathbb{R}^m,\boldsymbol{\mu}\in\mathbb{R}^M_+}{\operatorname{argmin}} \quad & -\mathcal{L}(\mathbf{x}_1^{k+1},\dots,\mathbf{x}_N^{k+1},\boldsymbol{\lambda},\boldsymbol{\mu}) \\ & +\frac{1}{2\rho}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^k\|_2^2+\frac{1}{2\rho}\|\boldsymbol{\mu}-\boldsymbol{\mu}^k\|_2^2.\end{aligned} \tag{2.9}$$

Similar to the convergence analysis of the PCPM algorithm in [7], we now establish some fundamental estimates of the distance at each iteration k between the solution point $(\mathbf{x}_1^{k+1},\dots,\mathbf{x}_N^{k+1},\boldsymbol{\lambda}^{k+1},\boldsymbol{\mu}^{k+1})$ and the saddle point $(\mathbf{x}_1^*,\dots,\mathbf{x}_N^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)$.

Proposition 2.2.3. Let $(\mathbf{x}_1^*,\dots,\mathbf{x}_N^*,\boldsymbol{\lambda}^*,\boldsymbol{\mu}^*)$ be a saddle point of the optimization problem (2.2). For all $k\geq 0$, we have that

$$\begin{aligned}\sum_{\mathrm{i}=1}^N\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^*\|_2^2 \leq & \sum_{\mathrm{i}=1}^N\|\mathbf{x}_\mathrm{i}^k-\mathbf{x}_\mathrm{i}^*\|_2^2-\sum_{\mathrm{i}=1}^N\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^k\|_2^2 \\ & +2\rho\Big[(\boldsymbol{\lambda}^*-\boldsymbol{\gamma}^{k+1})^T\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1}+\sum_{\mathrm{j}=1}^M(\mu_\mathrm{j}^*-\nu_\mathrm{j}^{k+1})\sum_{\mathrm{i}=1}^N g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1})\Big]\end{aligned} \tag{2.10}$$

and

$$\begin{aligned}&\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*\|_2^2+\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^*\|_2^2\leq\|\boldsymbol{\lambda}^k-\boldsymbol{\lambda}^*\|_2^2+\|\boldsymbol{\mu}^k-\boldsymbol{\mu}^*\|_2^2 \\ &-\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2-\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^{k+1}\|_2^2-\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^k\|_2^2-\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^k\|_2^2 \\ &+2\rho\Big[(\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k+1})^T\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^k+\sum_{\mathrm{j}=1}^M(\nu_\mathrm{j}^{k+1}-\mu_\mathrm{j}^{k+1})\sum_{\mathrm{i}=1}^N g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^k) \\ &\quad+(\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*)^T\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1}+\sum_{\mathrm{j}=1}^M(\mu_\mathrm{j}^{k+1}-\mu_\mathrm{j}^*)\sum_{\mathrm{i}=1}^N g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1})\Big]\end{aligned} \tag{2.11}$$

Proof. The details of the proof are provided in Section 2.6.1. □

Theorem 2.2.4 (Global Convergence)**.** Assume that Assumption 2.2.1 to Assumption 2.2.3 hold. Given a scalar $0 < \epsilon < 1$, choose a step size ρ satisfying

$$0 < \rho \leq \min \left\{ \frac{1-\epsilon}{A_{max} + ML_{max}}, \frac{1-\epsilon}{NA_{max}}, \frac{1-\epsilon}{NL_{max}} \right\}, \tag{2.12}$$

where $A_{max} := \max_{\mathrm{i}=1}^{N}\{\|A_\mathrm{i}\|_2\}$, and $L_{max} := \max_{\mathrm{j}=1}^{M}\left\{\max_{\mathrm{i}=1}^{N}\{L_{\mathrm{ji}}\}\right\}$.
Let $\{\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k\}$ be the sequence generated by Algorithm 1, with an arbitrary point $(\mathbf{x}_1^0, \ldots, \mathbf{x}_N^0, \boldsymbol{\lambda}^0, \boldsymbol{\mu}^0)$; then the sequence converges globally to a saddle point $(\mathbf{x}_1^* \ldots \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ of the optimization problem (2.2).

Proof. Please see Section 2.6.2 for details. □

To establish convergence rate, we need to make an additional assumption on Problem (2.2), as follows.

Assumption 2.2.4 (Lipschitz Inverse Mapping)**.** Assume that there exists a unique saddle point $(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ such that, given an inverse mapping $\mathcal{S}^{-1} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^M \to \mathbb{R}$:

$$\begin{aligned} &\mathcal{S}^{-1}(\mathbf{u}_1, \ldots, \mathbf{u}_N, \mathbf{v}) \\ &= \arg \min_{(\mathbf{x}_1,\ldots,\mathbf{x}_N)\in\prod_{\mathrm{i}=1}^{N}\mathcal{X}_\mathrm{i}} \max_{\boldsymbol{\lambda}\in\mathbb{R}^m, \boldsymbol{\mu}\in\mathbb{R}_+^M} \left\{ \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \sum_{\mathrm{i}=1}^{N} \mathbf{x}_\mathrm{i}^T \mathbf{u}_\mathrm{i} + \boldsymbol{\lambda}^T \mathbf{v} + \boldsymbol{\mu}^T \mathbf{w} \right\}, \end{aligned} \tag{2.13}$$

and a fixed positive real number $\tau > 0$, we have

$$\sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i} - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu} - \boldsymbol{\mu}^*\|_2^2 \leq a^2 \left(\sum_{\mathrm{i}=1}^{N} \|\mathbf{u}_\mathrm{i}\|_2^2 + \|\mathbf{v}\|_2^2 + \|\mathbf{w}\|_2^2 \right)$$

for some $a \geq 0$, whenever the point $(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathcal{S}^{-1}(\mathbf{u}_1, \ldots, \mathbf{u}_N, \mathbf{v}, \mathbf{w})$ and

$$\sum_{\mathrm{i}=1}^{N} \|\mathbf{u}_\mathrm{i}\|_2^2 + \|\mathbf{v}\|_2^2 + \|\mathbf{w}\|_2^2 \leq \tau^2.$$

The above assumption states that the inverse mapping $\mathcal{S}^{-1}$ is Lipschitz continuous at the origin with modulus a. By Proposition 2 in [16], to obtain a plausible condition for the

Lipschitz continuity of $\mathcal{S}^{-1}$ at the origin, we appeal to the *strong second-order conditions for optimality* which are comprised of the following properties:

(i) There is a saddle point $(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ of the optimization problem (2.2) such that $\mathbf{x}_\mathrm{i}^* \in \mathrm{i}nt(\mathcal{X}_\mathrm{i})$ for all $\mathrm{i} = 1 \ldots N$, where $\mathrm{i}nt(\mathcal{X}_\mathrm{i})$ denotes the interior of the convex set $\mathcal{X}_\mathrm{i}$. Moreover, for all $\mathrm{i} = 1 \ldots N$ and $\mathrm{j} = 1, \ldots, M$, function $g_{\mathrm{ji}}(\mathbf{x}_\mathrm{i})$ is twice continuously differentiable on a neighborhood of $\mathbf{x}_\mathrm{i}^*$.

(ii) Let $\mathcal{J}$ denote the set of active constraint indices at the point of $(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*)$: $\mathcal{J} = \Big\{\mathrm{j} = 1, \ldots, M \Big| \sum_{\mathrm{i}=1}^N g_{\mathrm{ji}}(\mathbf{x}_\mathrm{i}^*) = 0\Big\}$. Then $\mu_\mathrm{j}^* > 0$ for all $\mathrm{j} \in \mathcal{J}$, and $\Big\{\sum_{\mathrm{i}=1}^N \nabla_{\mathbf{x}_\mathrm{i}} g_{\mathrm{ji}}(\mathbf{x}_\mathrm{i}^*)\Big\}_{\mathrm{j}\in\mathcal{J}} \cup \Big\{\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big\}$ forms a linearly independent set.

(iii) The Hessian matrix $H \coloneqq \nabla^2_{(\mathbf{x}_1,\ldots,\mathbf{x}_N)}\mathcal{L}(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ satisfies $\mathbf{y}^T H \mathbf{y} > 0$ for any

$$\mathbf{y} \coloneqq \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{pmatrix} \in \left\{ \mathbf{y} \neq \mathbf{0} \,\middle|\, \begin{array}{l} \sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{y}_\mathrm{i} = \mathbf{0} \\ \sum_{\mathrm{i}=1}^N \mathbf{y}_\mathrm{i}^T \Big[\nabla_{\mathbf{x}_\mathrm{i}} g_{\mathrm{ji}}(\mathbf{x}_\mathrm{i}^*)\Big] = 0, \quad \forall \mathrm{j} \in \mathcal{J} \end{array} \right\}.$$

We now establish the linear convergence rate of Algorithm 1.

Theorem 2.2.5 (Linear Convergence Rate)**.** Assume that Assumption 2.2.1 to Assumption 2.2.4 hold. Let ρ satisfy (2.12) and let $\{\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k\}$ be a sequence generated by Algorithm 1, with an arbitrary starting point $(\mathbf{x}_1^0, \ldots, \mathbf{x}_N^0, \boldsymbol{\lambda}^0, \boldsymbol{\mu}^0)$; then the sequence converges linearly to the unique saddle point $(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$. More specifically, there exists an integer $\bar{k}$ such that, for all $k \geq \bar{k}$, we have:

$$\begin{aligned} &\sum_{\mathrm{i}=1}^N \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^*\|_2^2 \\ \leq &\theta^2\Big(\sum_{\mathrm{i}=1}^N \|\mathbf{x}_\mathrm{i}^k - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^*\|_2^2\Big), \end{aligned} \tag{2.14}$$

where $0 < \theta < 1$.

Proof. Please see Section 2.6.3 for details. □

2.2.3 Numerical Experiments

Consider the following 20-dimensioned, block-separable convex optimization problem with 17 nonlinear coupling constraints, modeling a decentralized planning of an economic system and suggested by [17]:

$$\begin{aligned}
\underset{x_1,\ldots,x_{20}}{\text{minimize}}\quad & x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2 \\
& +2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + (x_{11}-9)^2 \\
& +10(x_{12}-1)^2 + 5(x_{13}-7)^2 + 4(x_{14}-14)^2 + 27(x_{15}-1)^2 + x_{16}^4 \\
& +(x_{17}-2)^2 + 13(x_{18}-2)^2 + (x_{19}-3)^2 + x_{20}^2 + 95 \\
\text{subject to}\quad & 3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120 \le 0 \\
& 5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40 \le 0 \\
& \frac{1}{2}(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30 \le 0 \\
& x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \le 0 \\
& 4x_1 + 5x_2 - 3x_7 + 9x_8 - 105 \le 0 \\
& 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0 \\
& 3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10} \le 0 \\
& -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0 \\
& x_1 + x_2 + 4x_{11} - 21x_{12} \le 0 \\
& x_1^2 + 15x_{11} - 8x_{12} - 28 \le 0 \\
& 4x_1 + 9x_2 + 5x_{13}^2 - 9x_{14} - 87 \le 0 \\
& 3x_1 + 4x_2 + 3(x_{13}-6)^2 - 14x_{14} - 10 \le 0 \\
& 14x_1^2 + 35x_{15} - 79x_{16} - 92 \le 0 \\
& 15x_2^2 + 11x_{15} - 61x_{16} - 54 \le 0 \\
& 5x_1^2 + 2x_2 + 9x_{17}^4 - x_{18} - 68 \le 0 \\
& x_1^2 - x_2 + 19x_{19} - 20x_{20} + 19 \le 0 \\
& 7x_1^2 + 5x_2^2 + x_{19}^2 - 30x_{20} \le 0 \qquad (2.15)
\end{aligned}$$

A minimum function value of 133.723 can be obtained at the point of (2.18, 2.35, 8.77, 5.07, 0.99, 1.43, 1.33, 9.84, 8.29, 8.37, 2.28, 1.36, 6.08, 14.17, 1.00, 0.66, 1.47, 2.00, 1.05, 2.06).[1] Applying Algorithm 1 and decomposing the original problem into 19 small sub-problems, we achieve the following convergence results, presented in Figure 2.1.

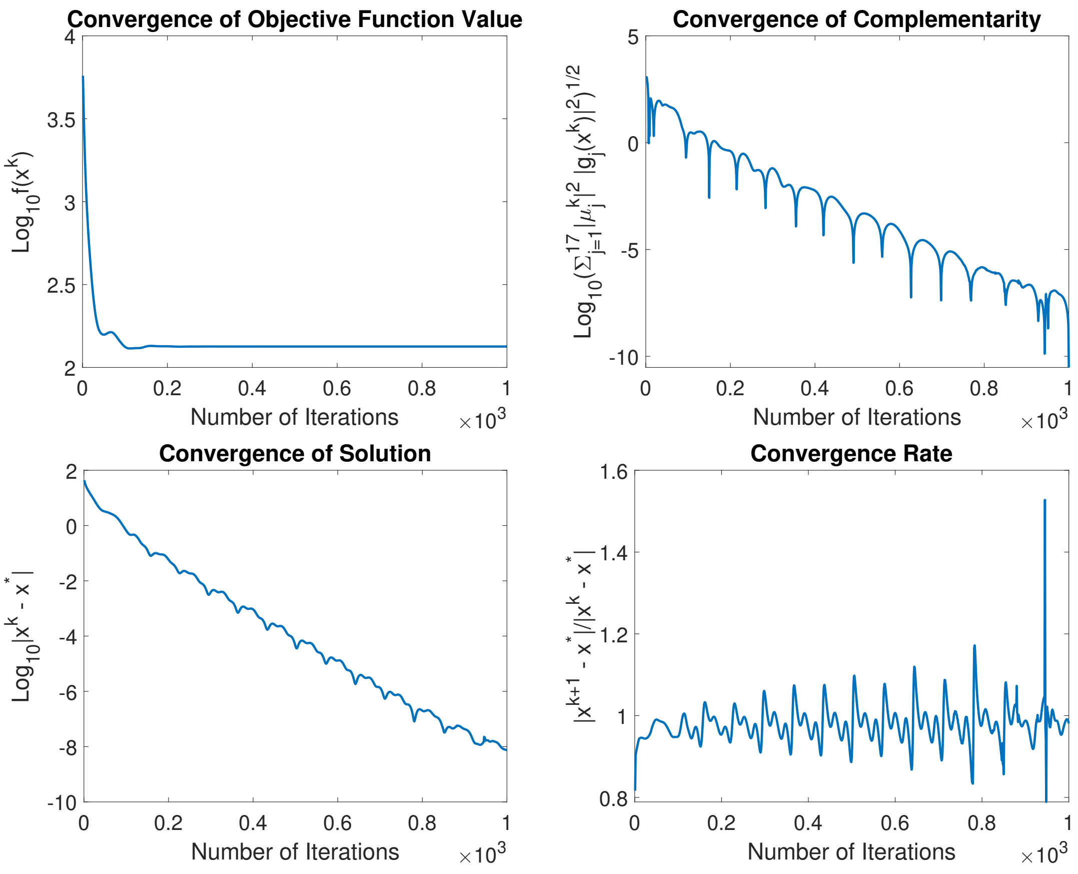


Figure 2.1. Convergence Results of Applying Algorithm 1 to Problem (2.15) with $\rho = 0.009$.

Next, by replacing the term of x_{16}^4 in the objective function of (2.15) with x_{16}^2 and the term of x_{17}^4 in the 15-th constraint with x_{17}^2, we make the modified problem satisfy Assumption 2.2.4. A minimum function value of 133.687 can be obtained at the point of (2.18, 2.34, 8.76, 5.07, 0.99, 1.43, 1.34, 9.84, 8.30, 8.36, 2.27, 1.36, 6.08, 14.17, 1.00, 0.64, 2.00, 2.00, 1.04, 2.06). An additional linear convergence rate of applying Algorithm 1 is observed in the following convergence results, presented in Figure 2.2.

[1]The solution is obtained using the nonlinear constrained optimization solver **filter** of Neos Solver at https://neos-server.org/neos/solvers/.

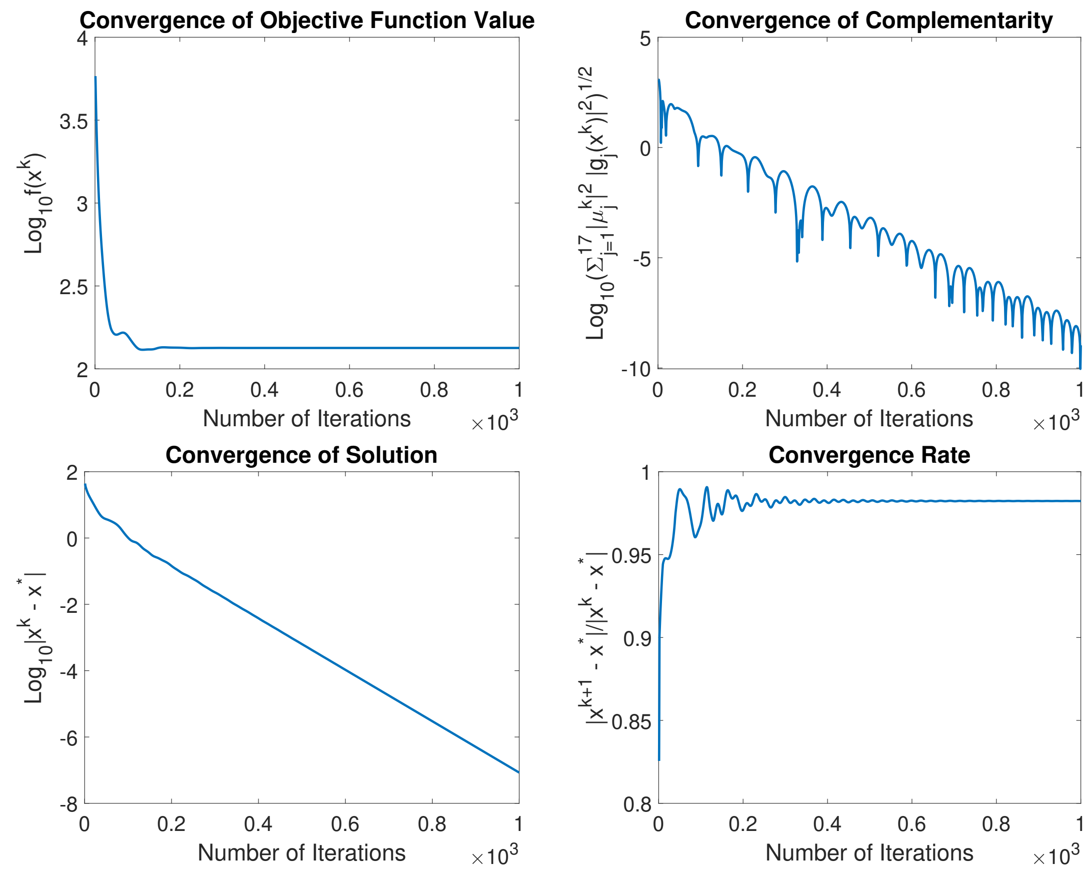


Figure 2.2. Convergence Results of Applying Algorithm 1 to the Modified Problem (2.15) with $\rho = 0.009$.

2.3 Extending the N-block PCPM Algorithm to an Asynchronous Scheme

As a starting point, we first consider the following N-block convex optimization problem with only linear coupling constraints:

$$\begin{aligned} \underset{\mathbf{x}_1,\dots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) \\ \text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \mathrm{i} = 1 \dots N, \\ & \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} = \mathbf{b}. \end{aligned} \tag{2.16}$$

The decision variables, the objective function and the constraints are the same as in the optimization problem (2.1). When applying Algorithm 1 to solving the above problem, each iteration can be interpreted as main-worker paradigm [18], shown in Figure 2.3. At

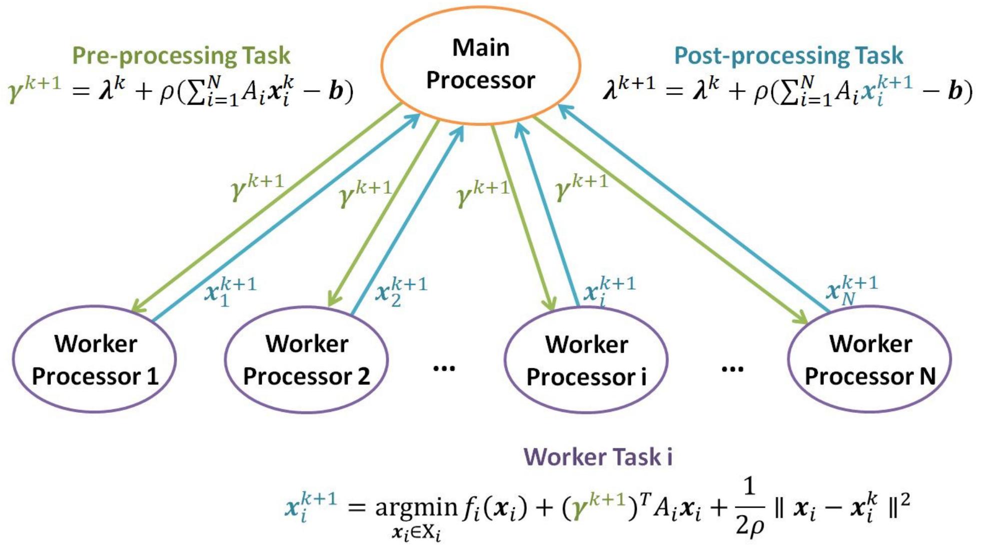

Main Processor
Pre-processing Task
$\gamma^{k+1} = \lambda^k + \rho(\sum_{i=1}^N A_i x_i^k - b)$
Post-processing Task
$\lambda^{k+1} = \lambda^k + \rho(\sum_{i=1}^N A_i x_i^{k+1} - b)$
γ^{k+1}
x_1^{k+1}
x_2^{k+1}
x_i^{k+1}
x_N^{k+1}
Worker Processor 1
Worker Processor 2
...
Worker Processor i
...
Worker Processor N
Worker Task i
$x_i^{k+1} = \operatorname{argmin}_{x_i \in X_i} f_i(x_i) + (\gamma^{k+1})^T A_i x_i + \frac{1}{2\rho} \| x_i - x_i^k \|^2$

Figure 2.3. Main-Worker Paradigm for Algorithm 1.

each iteration k, a predictor update of $\boldsymbol{\gamma}^{k+1}$ is first performed on a *main processor* and is broadcast to each *worker processor*, which is called a *pre-processing task*. Upon receiving the updated predictor variable from the main processor, each worker processor solves the decomposed sub-problem in parallel and send its updated primal decision variable $\mathbf{x}_\mathrm{i}^{k+1}$ back to the main processor, which is called a *worker task*. After gathering all updated decision variables, a corrector update is then performed on the main processor, which is called a *post-processing task*.

The speed of the algorithm is significantly limited by the slowest worker processor, since the post-processing task can not start until all worker tasks are finished and the results are sent back to the main processor. For large-scale problems, with the number of worker processors increasing, the issue of node synchronization can be a major concern for the performance of synchronous distributed algorithms. While in an asynchronous scheme, the main processor can proceed with only part of worker tasks finished. Figure 2.4 shows an example of 1 main processor and 4 worker processors with different lengths of computation and communication delays. In the asynchronous scheme, the main processor starts a new iteration whenever receives the results from at least 2 worker processors, which leads to much faster iterations than the synchronous scheme.

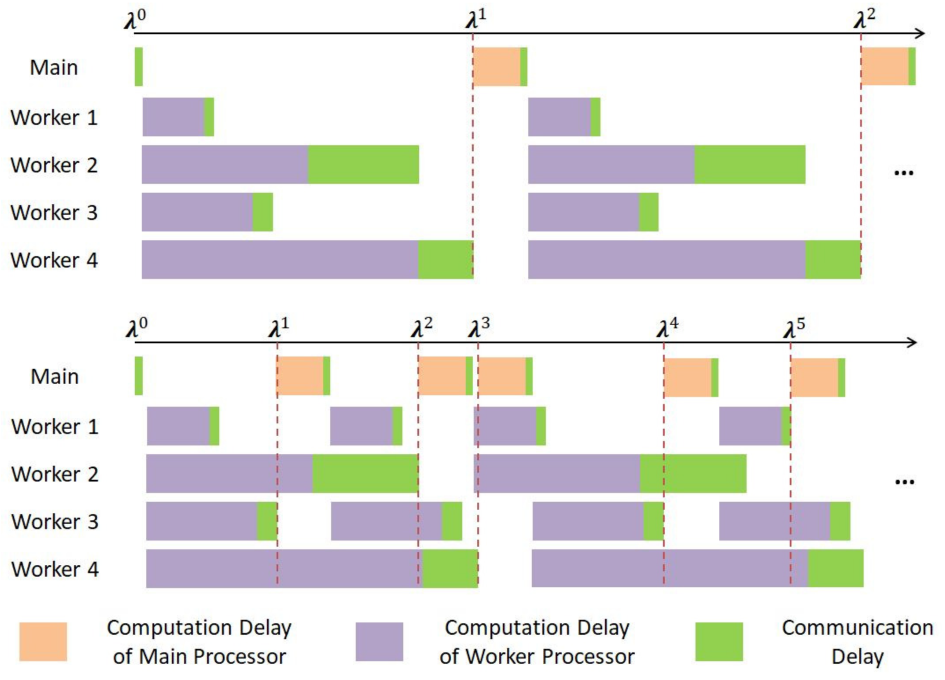


Figure 2.4. Illustration of how asynchronous scheme iterates faster than synchronous scheme.

In this section, we extend the N-block PCPM algorithm to an asynchronous scheme to solve the linearly constrained N-block convex optimization problem (2.16).

2.3.1 Asynchronous N-block PCPM Algorithm for Convex Optimization Problems with Linear Coupling Constraints

To achieve the convergence of the asynchronous N-block PCPM algorithm, similar to [19], [20], we require that the asynchronous delay of each parallel worker processor is bounded. Let $k \geq 0$ denote the iteration index on the main processor. At each iteration k, let $\mathcal{A}_k \subseteq \{1, \dots, N\}$ denote the subset of worker processors from whom the main processor receives the updated decision variable $\hat{\mathbf{x}}_\text{i}$, and let $\mathcal{A}_k^\complement \subseteq \{1, \dots, N\}$ denote the rest of the worker processors, whose information does not arrive.

Definition 2.3.1 (Bounded Delay)**.** Let an integer $\tau \geq 1$ denote the maximum tolerable delay. At any iteration $k \geq 0$, with a *bounded delay*, it must holds that $\text{i} \in \mathcal{A}_k \cup \mathcal{A}_{k-1} \cup \cdots \cup \mathcal{A}_{k-\tau+1}$ for all $\text{i} = 1, \dots, N$. When $\tau = 1$, it's a synchronous scheme.

At each each iteration k on the main processor, all worker processors are divided into two sets $\mathcal{A}_k$ and $\mathcal{A}_k^\complement$, distinguished by whether their information arrives or not at the current moment. Let d_i denote the number of iterations that each worker processor i is delayed. If $d_\mathrm{i} < \tau - 1$ for each worker processor $\mathrm{i} \in \mathcal{A}_k^\complement$, the main processor uses the partially updated decision variables $(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1})$ to perform both corrector and predictor updates for the Lagrangian multiplier; otherwise, the main processor must wait until the worker processors with $d_\mathrm{i} = \tau - 1$ finish their tasks with information received by the master processor. Consequently, new divide of worker processors into $\mathcal{A}_k$ and $\mathcal{A}_k^\complement$ is generated, and the bounded delay condition is then satisfied. The overall structure of the Asynchronous N-Block PCPM algorithm for solving the linearly constrained convex optimization problem (2.16) is presented in Algorithm 2 and Algorithm 3.

Algorithm 2 AN-PCPM (Main Processor)

```
1: **Initialization** choose an arbitrary starting point $\boldsymbol{\lambda}^0$.
2: $k \leftarrow 0,\ d_1, d_2, \ldots, d_N \leftarrow 0$
3: **wait** until receiving $\{\hat{\mathbf{x}}_\mathrm{i}\}_{\mathrm{i}=1\ldots N}$
4: **update**:
       $$\mathbf{x}_\mathrm{i}^0 = \hat{\mathbf{x}}_\mathrm{i}, \quad \mathrm{i} = 1 \ldots N \qquad (2.17)$$
5: **broadcast** $\hat{\boldsymbol{\gamma}} = \boldsymbol{\lambda}^0 + \rho(\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^0 - \mathbf{b})$ to all worker processors
6: **while** termination conditions are not met **do**
7:     **wait** until receiving $\{\hat{\mathbf{x}}_\mathrm{i}\}_{\mathrm{i}\in\mathcal{A}_k}$ such that $d_\mathrm{i} < \tau, \quad \forall \mathrm{i} \in \mathcal{A}_k^\complement$
8:     **update**:
           $$\mathbf{x}_\mathrm{i}^{k+1} = \begin{cases} \hat{\mathbf{x}}_\mathrm{i}, & \forall \mathrm{i} \in \mathcal{A}_k \\ \mathbf{x}_\mathrm{i}^k, & \forall \mathrm{i} \in \mathcal{A}_k^\complement \end{cases} \qquad d_\mathrm{i} = \begin{cases} 0, & \forall \mathrm{i} \in \mathcal{A}_k \\ d_\mathrm{i} + 1, & \forall \mathrm{i} \in \mathcal{A}_k^\complement \end{cases} \qquad (2.18)$$
9:     **update**:
           $$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho\Big(\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{b}\Big) \qquad (2.19)$$
10:    **broadcast** $\hat{\boldsymbol{\gamma}} = \boldsymbol{\lambda}^{k+1} + \rho\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1}$ to the worker processors in $\mathcal{A}_k$
11:    $k++$
12: **return** $(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k)$
```

Algorithm 3 AN-PCPM (Woker Processor)

```
1: send $\hat{\mathbf{x}}_i = \mathbf{x}_i^0$ to the main processor
2: while not receiving termination signal do
3:     wait until receiving $\hat{\gamma}$
4:     calculate:
           $$\mathbf{y}_i = \underset{\mathbf{x}_i \in \mathcal{X}_i}{\operatorname{argmin}} f_i(\mathbf{x}_i) + \hat{\gamma}^T A_i \mathbf{x}_i + \frac{1}{2\rho}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2 \qquad (2.20)$$
5:     update: $\hat{\mathbf{x}}_i = \mathbf{y}_i$
6:     send $\hat{\mathbf{x}}_i$ to the main processor
```

2.3.2 Convergence Analysis

Different from the synchronous N-block PCPM algorithm, the convexity of f_i is not enough to achieve the global convergence due to the asynchronous delay in the system. We make the following additional assumption on problem (2.16).

Assumption 2.3.1 (Strong Convexity)**.** For all $i = 1 \dots N$, each $f_i : \mathcal{X}_i \to \mathbb{R}$ is a continuous, strongly convex function with modulus $\sigma_i > 0$.

Accordingly, we extend Lemma 2.2.1 to functions with strong convexity.

Lemma 2.3.1 (Inequality of Proximal Minimization Point with Strong Convexity)**.** Given a closed, convex set $\mathbb{Z} \subset \mathbb{R}^n$, and a continuous, strongly convex function $F : \mathbb{Z} \to \mathbb{R}$ with modulus σ. With a given point $\bar{\mathbf{z}} \in \mathbb{Z}$ and a positive number $\rho > 0$, if $\widehat{\mathbf{z}}$ is a proximal minimization point; i.e. $\widehat{\mathbf{z}} := \arg\min_{\mathbf{z} \in \mathbb{Z}} F(\mathbf{z}) + \frac{1}{2\rho}\|\mathbf{z} - \bar{\mathbf{z}}\|_2^2$, then we have that

$$F(\widehat{\mathbf{z}}) - F(\mathbf{z}) \le \frac{1}{2\rho}\|\bar{\mathbf{z}} - \mathbf{z}\|_2^2 - (\frac{\sigma}{2} + \frac{1}{2\rho})\|\widehat{\mathbf{z}} - \mathbf{z}\|_2^2 - \frac{1}{2\rho}\|\widehat{\mathbf{z}} - \bar{\mathbf{z}}\|_2^2, \quad \forall \mathbf{z} \in \mathbb{Z}. \qquad (2.21)$$

Proof. Denote $\Phi(\mathbf{z}) = F(\mathbf{z}) + \frac{1}{2\rho}\|\mathbf{z} - \bar{\mathbf{z}}\|_2^2$. By the definition of $\widehat{\mathbf{z}}$, we have $\partial_{\mathbf{z}}\Phi(\widehat{\mathbf{z}}) = \mathbf{0}$. Since $\Phi(\mathbf{z})$ is strongly convex with modulus $\sigma + \frac{1}{\rho}$, it follows that $\Phi(\mathbf{z}) - \Phi(\widehat{\mathbf{z}}) \ge (\frac{\sigma}{2} + \frac{1}{2\rho})\|\widehat{\mathbf{z}} - \mathbf{z}\|_2^2$ for any $\mathbf{z} \in \mathbb{Z}$. □

Now, we present the main convergence result.

Theorem 2.3.2 (Sub-linear Convergence Rate)**.** Let Assumption 2.2.1, Assumption 2.2.3 and Assumption 2.3.1 hold. Choose a step size ρ satisfying:

$$\rho \leq \frac{\sigma_{min}}{25N(\tau-1)^2 A_{max}}, \tag{2.22}$$

where $\sigma_{min} := \min_{\mathrm{i}=1}^{N}\{\sigma_\mathrm{i}\}$. Denote $\bar{\mathbf{x}}_\mathrm{i}^k = \frac{1}{k}\sum_{k'=1}^{k}\mathbf{x}_\mathrm{i}^{k'}$ for all $\mathrm{i} = 1, \ldots, N$, where $\{(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k)\}$ is the sequence generated by Algorithm 2 and Algorithm 3, then for all $k > 0$, it holds that:

$$\left|\sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\bar{\mathbf{x}}_\mathrm{i}^k) - \sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^*)\right| \leq \frac{\delta_{\boldsymbol{\lambda}} C_1 + C_2}{k}, \qquad \left\|\sum_{\mathrm{i}=1}^{N} A_\mathrm{i}\bar{\mathbf{x}}^k - \mathbf{b}\right\|_2 \leq \frac{C_1}{k}, \tag{2.23}$$

where $\delta_{\boldsymbol{\lambda}} = \|\boldsymbol{\lambda}^*\|_2$ and C_1, C_2 are some finite constants.

Proof. Please see Section 2.7.1 for details. □

2.4 Numerical Experiments

2.4.1 An Optimization Problem on a Graph

In this subsection, we consider an optimization problem on a graph, arising from the training process of regressors with spatial clustering, proposed by [11]. Traditional regressors obtains a parameter vector $\mathbf{x}$ via solving the following optimization problem on a training data set:

$$\underset{\mathbf{x}\in\mathcal{X}}{\text{minimize}} \quad \sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\mathbf{x}) + r(\mathbf{x}), \tag{2.24}$$

where N is the number of data points, $\mathcal{X} \subset \mathbb{R}^n$ describes the constraints on the parameter vector, each function $f_\mathrm{i} : \mathbb{R}^n \to \mathbb{R}$ denotes the loss function on each training data point for all $\mathrm{i} = 1, \ldots, N$, and $r : \mathbb{R}^n \to \mathbb{R}$ denotes some type of regularization function.

When the spatial information is accessible, such as the latitude and longitude data, a map of data points then becomes available. Instead of using a global regressor with a common parameter vector $\mathbf{x}$ for the whole data set, a local regressor can be built at each data point $\mathrm{i} = 1, \ldots, N$ with a local parameter vector $\mathbf{x}_\mathrm{i} \in \mathbb{R}^n$. Let d_{ij} denote the distance between the data point i and j ($\mathrm{i} \neq \mathrm{j}$). Different from distributed learning, where a consensus constraint should be satisfied for all local variables, we require that the difference between two local parameter vectors $\|\mathbf{x}_\mathrm{i} - \mathbf{x}_\mathrm{j}\|_2^2$ decreases as the distance d_ij decreases. Let $\mathcal{N}_\epsilon(\mathrm{i})$ denote a set of data points within a neighborhood of point i, i.e., $\mathcal{N}_\epsilon(\mathrm{i}) := \{\mathrm{j} = 1, \ldots, N | \mathrm{j} \neq \mathrm{i}, d_\mathrm{ij} \leq \epsilon\}$. If any data point is regarded as a vertex and any two data points within a neighborhood are connected through an edge, a graph is then constructed as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of vertices with $N = |\mathcal{V}|$ and $\mathcal{E}$ denotes the set of edges with $p = |\mathcal{E}|$. Consider the following optimization problem on the graph:

$$\begin{aligned} \underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}\in\mathcal{V}} \Big[f_\mathrm{i}(\mathbf{x}_\mathrm{i}) + r(\mathbf{x}_\mathrm{i}) \Big] + \omega \sum_{(\mathrm{j},k)\in\mathcal{E}} w_{\mathrm{j}k} \|\mathbf{x}_\mathrm{j} - \mathbf{x}_k\|_2^2 \\ \text{subject to} \quad & \mathbf{x}_\mathrm{i} \in \mathcal{X}_\mathrm{i}, \quad \mathrm{i} = 1, \ldots, N. \end{aligned} \tag{2.25}$$

Different from [11], we use $\|\cdot\|_2^2$ instead of $\|\cdot\|_2$. The parameter $w_{\mathrm{j}k}$ along each edge $(\mathrm{j}, k) \in \mathcal{E}$, describing the weight of the penalty term of the difference between the two connected vertices, increases as $d_{\mathrm{j}k}$ decreases. The global parameter ω describes the trade-off between minimizing the individual loss function on each data point and agreeing with neighbors. When $\omega = 0$, $\mathbf{x}_\mathrm{i}^*$ is simply the solution to the optimization problem: $\underset{\mathbf{x}_\mathrm{i}\in\mathbb{X}_\mathrm{i}}{\text{minimize}} \quad f_\mathrm{i}(\mathbf{x}_\mathrm{i}) + r(\mathbf{x}_\mathrm{i})$, obtained locally at each vertex i. When $\omega \to +\infty$, the model reduces to a traditional regressor without spatial clustering.

Once the optimal solution $(\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*)$ is obtained, for any new node i′, the local regressor can be evaluated with the local parameter vector $\mathbf{x}_{\mathrm{i}'}$, estimated through the interpolation of the solution:

$$\underset{\mathbf{x}_{\mathrm{i}'}\in\mathcal{X}_{\mathrm{i}'}}{\text{minimize}} \quad \sum_{\mathrm{j}\in\mathcal{N}_\epsilon(\mathrm{i}')} w_{\mathrm{i}'\mathrm{j}} \|\mathbf{x}_{\mathrm{i}'} - \mathbf{x}_\mathrm{j}^*\|_2^2. \tag{2.26}$$

2.4.2 Two Problem Reformulations

To apply N-block PCPM algorithm to solve the graph optimization problem (2.25), we first need to reformulate it into the block-separable form as in (2.16).

Similar to [11], for each pair of connected vertices $(\mathbf{x}_\mathrm{j}, \mathbf{x}_k)$ along the edge $(\mathrm{j}, k) \in \mathcal{E}$, introducing a copy $(\mathbf{z}_{\mathrm{j}k}, \mathbf{z}_{k\mathrm{j}})$, we can rewrite the graph optimization problem (2.25) as

- **Problem Reformulation 1**

$$\begin{aligned}
\underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}\in\mathcal{V}} \Big[f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) + r(\mathbf{x}_{\mathrm{i}})\Big] + \omega \sum_{(\mathrm{j},k)\in\mathcal{E}} w_{\mathrm{j}k}\|\mathbf{z}_{\mathrm{j}k} - \mathbf{z}_{k\mathrm{j}}\|_2^2 \\
\text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \forall \mathrm{i} \in \mathcal{V}, \\
& \mathbf{x}_{\mathrm{i}} - \mathbf{z}_{\mathrm{ij}} = \mathbf{0}, \quad \forall \mathrm{j} \in \mathcal{N}_\epsilon(\mathrm{i}), \quad \forall \mathrm{i} \in \mathcal{V}.
\end{aligned} \tag{2.27}$$

The reformulated problem can be decomposed into N sub-problems on vertices and p sub-problems along edges, using N-block PCPM algorithm. One small issue is that the objective function of the edge sub-problem, $\|\mathbf{z}_{\mathrm{j}k} - \mathbf{z}_{k\mathrm{j}}\|_2^2$, is not strongly convex.

To overcome this limitation, we propose an alternative way of rewriting (2.25). For each edge $(\mathrm{j}, k) \in \mathcal{E}$, introducing a slack variable $\mathbf{z}_{\mathrm{j}k} = \mathbf{x}_{\mathrm{j}} - \mathbf{x}_k$, we can rewrite the graph optimization problem (2.25) as

- **Problem Reformulation 2**

$$\begin{aligned}
\underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}\in\mathcal{V}} \Big[f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) + r(\mathbf{x}_{\mathrm{i}})\Big] + \omega \sum_{(\mathrm{j},k)\in\mathcal{E}} w_{\mathrm{j}k}\|\mathbf{z}_{\mathrm{j}k}\|_2^2 \\
\text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{X}_{\mathrm{i}}, \quad \forall \mathrm{i} \in \mathcal{V}, \\
& \mathbf{x}_{\mathrm{j}} - \mathbf{x}_k - \mathbf{z}_{\mathrm{j}k} = \mathbf{0}, \quad \forall (\mathrm{j}, k) \in \mathcal{E}.
\end{aligned} \tag{2.28}$$

The reformulated problem can also be decomposed into N sub-problems on vertices and p sub-problems along edges. However, in this way of rewriting (2.25), the objective function of each decomposed sub-problem enjoys the nice property of strong convexity. When applying N-block PCPM algorithm, a linear convergence rate is expected for synchronous iteration scheme, and a sub-linear convergence rate is expected for an asynchronous scheme.

2.4.3 Housing Price Prediction

In this subsection, we present an application example of the graph optimization problem, where the housing price is predicted based on a set of features, including the number of bedrooms, the number of bathrooms, the number of square feet, and the latitude and longitude of each house. We use the same data set as [11], a list of 985 real estate transactions over a period of one week during May of 2008 in the Greater Sacramento area. All data is standardized with zero mean and unit variance, and all missing data is then set to zero. We randomly select a subset of 193 transactions as our test data set, and use the rest as our training data set.

The graph is constructed based on the latitude and longitude of each house. The rule of selecting neighbors is slightly different from [11]. For each house, we connect it with all the other houses within a distance of 1.0 mile. If the number of connected houses is less then 5, we connect more nearest houses until the number of neighbors reaches 5. The resulting graph has 792 vertices and 4303 edges. Thus, the graph optimization problem can be decomposed into $792 + 4303 = 5095$ sub-problems, using N-block PCPM algorithm.

At each data point $\mathrm{i} = 1 \dots 792$, the decision variable is $\mathbf{x}_{\mathrm{i}} = (x_{\mathrm{i}0}, x_{\mathrm{i}1}, x_{\mathrm{i}2}, x_{\mathrm{i}3})$. The predicted price for each house is:

$$x_{\mathrm{i}0} + x_{\mathrm{i}1} \times (\mathrm{num\_bed})_{\mathrm{i}} + x_{\mathrm{i}2} \times (\mathrm{num\_bath})_{\mathrm{i}} + x_{\mathrm{i}3} \times (\mathrm{num\_sq\_ft})_{\mathrm{i}},$$

where $(\mathrm{num\_bed})_{\mathrm{i}}$, $(\mathrm{num\_bath})_{\mathrm{i}}$ and $(\mathrm{num\_sq\_ft})_{\mathrm{i}}$ are the number of bedrooms, the number of bathrooms and the number of square feets for each house respectively. At each vertex i, the objective function

$$f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) = \|x_{\mathrm{i}0} + x_{\mathrm{i}1} \times (\mathrm{num\_bed})_{\mathrm{i}} + x_{\mathrm{i}2} \times (\mathrm{num\_bath})_{\mathrm{i}} + x_{\mathrm{i}3} \times (\mathrm{num\_sq\_ft})_{\mathrm{i}} - (\mathrm{price})_{\mathrm{i}}\|_2^2$$

is strongly convex, as well as the regularization function

$$r(\mathbf{x}_{\mathrm{i}}) = \mu\Big(\|x_{\mathrm{i}1}\|_2^2 + \|x_{\mathrm{i}2}\|_2^2 + \|x_{\mathrm{i}3}\|_2^2\Big),$$

where $(\mathrm{price})_{\mathrm{i}}$ is the actual sales price for each house, and μ is a constant regularization parameter, fixed as $\mu = 0.1$.

2.4.4 Numerical Results of Synchronous N-block PCPM Algorithm

We first apply Algorithm 1 to solve the two reformulated problem (2.27) and (2.28), and compare the performance. The convergence results are shown in Figure 2.5. Due to the

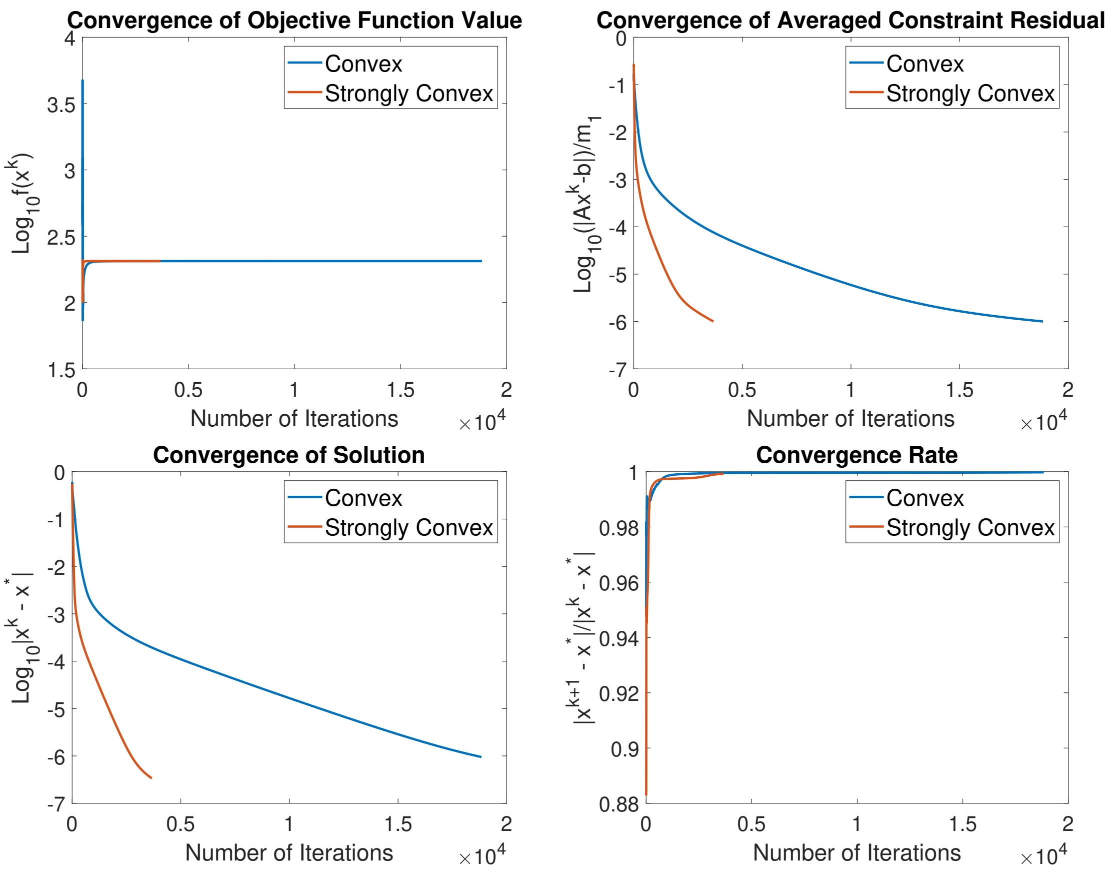


Figure 2.5. Convergence Results of Applying Algorithm 1 to solve Reformulated Problems (2.27) and (2.28) with $\omega = 1.0$ and $\rho = 0.06$.

strong convexity of the reformulated problem (2.28), the algorithm converges much faster than solving the reformulated problem (2.27).

We also plot the mean square error (MSE) on the testing data set using various values of ω, shown in Figure 2.6. When $\omega = 1.0$, a minimum MSE of 0.27 can be obtained on the testing data set. We fixed $\omega = 1.0$ for all the numerical experiments.

2.4.5 Numerical Results of Asynchronous N-block PCPM Algorithm

We apply Algorithm 2 and Algorithm 3 to solve the reformulated problem (2.28) with a maximum delay $\tau = 4$. The convergence results of are shown in Figure 2.7. A sub-linear convergence rate is observed.

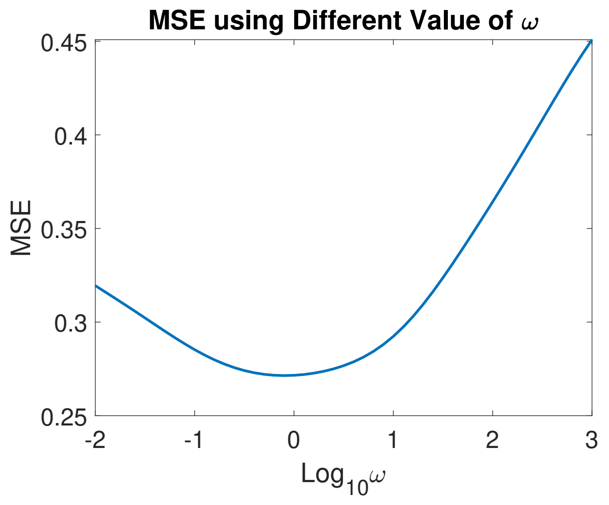


Figure 2.6. MSE for Testing Data Set with ω Varying from 10^{-2} to 10^{3} and $\mu = 0.1$.

While implementing the algorithm as a sequential code, we simulate the elapsed wall-clock time on the main processor. As illustrated in Figure 2.4, the computation delay of the main processor is set as 1.0 second, the computation delay of each worker processor for solving vertex sub-problem is set as 1.2 second, the computation delay of each worker processor for solving edge sub-problem is set as 0.6 second, and the communication delay of each worker processor is uniformly drawn from a range of 0.0 to 1.0 second. Under these settings, we simulate the elapsed wall-clock time on the main processor for the different values of maximum delay $\tau = 1, 2, 4, 7$, shown in Figure 2.8. We observe that, with a larger number of maximum delay, the number of iterations, used for the asynchronous algorithm to converge, increases but the simulated elapsed wall-clock time decreases, which implies a faster convergence with more short-time iterations.

2.5 Conclusion and Future Works

In this chapter, we first proposed an N-block PCPM algorithm to solve N-block convex optimization problems with both linear and nonlinear constraints, with global convergence established. A linear convergence rate under the strong second-order conditions for optimality is observed in the numerical experiments. Next, for a starting point, we proposed an asynchronous N-block PCPM algorithm to solve linearly constrained N-block convex

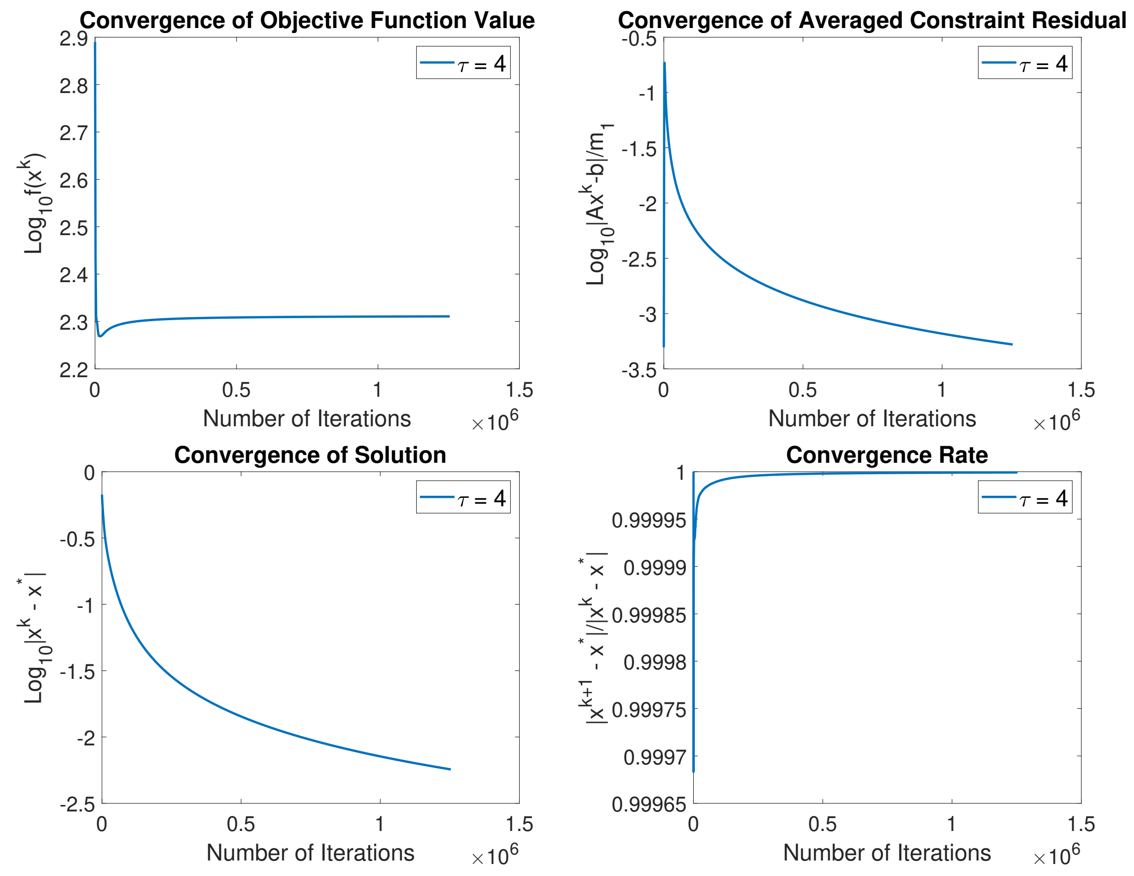


Figure 2.7. Convergence Results of Applying Algorithm 2 and Algorithm 3 to Solve the Reformulated Problem (2.28) with $\tau = 4$ and $\rho = 0.0005$.

optimization problems. The numerical results demonstrate the sub-linear convergence rate under the bounded delay assumption, as well as the faster convergence with more short-time iterations than a synchronous iterative scheme.

However, the performance of real asynchronous implementation of N-block PCPM algorithm is unknown, and thus in the future, more experiments (probably with much larger problem sizes) will be conducted on a multi-node computer cluster using MPI functions without blocking communication, such as MPI_Isend and MPI_Irecv. Also, the extension of the asynchronous N-block PCPM algorithm to solve N-block convex optimization problems with both linear and nonlinear constraints is worth to be explored.

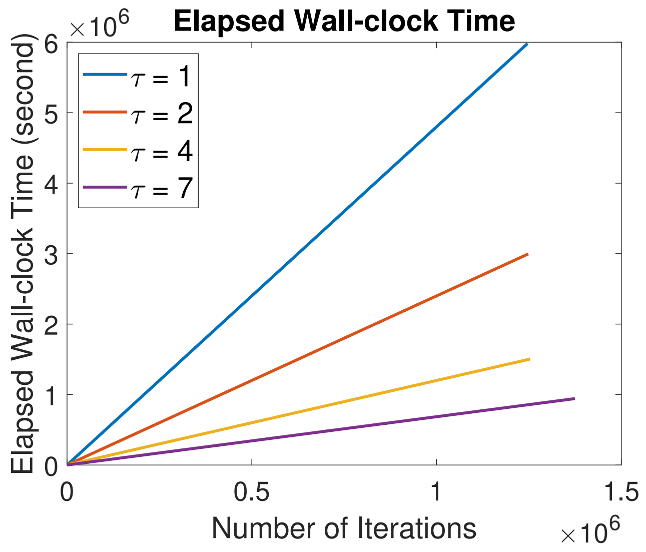


Figure 2.8. Simulated Elapsed Wall-clock Time on the Main Processor with Various Maximum Delay τ and a Same $\rho = 0.0005$.

2.6 Proofs in Section 2.2.2

2.6.1 Proof of Proposition 2.2.3

We first prove the inequality (2.10). From the primal minimization step (2.4), we know that $(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1})$ is the unique proximal minimization point of the Lagrangian function evaluated at the predictor variable: $\mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_N, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})$. Applying Lemma 2.2.1 with $\widehat{\mathbf{z}} = (\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1})$, $\bar{\mathbf{z}} = (\mathbf{x}_1^{k}, \dots, \mathbf{x}_N^{k})$ and $\mathbf{z} = (\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*})$, we have:

$$\begin{aligned} &2\rho\Big[\mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1}) - \mathcal{L}(\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*}, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})\Big] \\ &\le \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2 - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2 - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{k}\|_2^2. \end{aligned} \tag{2.29}$$

Since $(\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is a saddle point of the Lagrangian function $\mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_N, \boldsymbol{\lambda}, \boldsymbol{\mu})$, i.e., $\mathcal{L}(\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*}, \mathbf{p}^{k+1}) \le \mathcal{L}(\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*}, \boldsymbol{\lambda}^*) \le \mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^*)$, we have:

$$2\rho\Big[\mathcal{L}(\mathbf{x}_1^{*}, \dots, \mathbf{x}_N^{*}, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1}) - \mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\Big] \le 0. \tag{2.30}$$

Adding the above two inequalities yields the inequality (2.10) in Proposition 2.2.3.

To prove the second inequality, (2.11), in Proposition 2.2.3, we use a similar approach as above. By Lemma 2.2.2, we know that $(\boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})$ is the unique proximal minimization point of the function $-\mathcal{L}(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}, \boldsymbol{\mu})$. Applying Lemma 2.2.1 with $\widehat{\mathbf{z}} = (\boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})$, $\bar{\mathbf{z}} = (\boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ and $\mathbf{z} = (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$, we have:

$$\begin{aligned}
&2\rho\Big\{\Big[-\mathcal{L}(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})\Big] - \Big[-\mathcal{L}(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})\Big]\Big\}\\
\leq&\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^{k+1}\|_2^2\\
-&\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 - \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 - \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^k\|_2^2. \qquad (2.31)
\end{aligned}$$

By Lemma 2.2.2, we also know that $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$ is the unique proximal minimization point of the function $-\mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\mu})$. Applying Lemma 2.2.1 with $\widehat{\mathbf{z}} = (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$, $\bar{\mathbf{z}} = (\boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ and $\mathbf{z} = (\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$, we have:

$$\begin{aligned}
&2\rho\Big\{\Big[-\mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})\Big] - \Big[-\mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\Big]\Big\}\\
\leq&\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^*\|_2^2\\
-&\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^*\|_2^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|_2^2. \qquad (2.32)
\end{aligned}$$

Adding the above two inequalities yields the inequality (2.11) in Proposition 2.2.3.

2.6.2 Proof of Theorem 2.2.4

By adding the two inequalities (2.10) and (2.11) in Proposition 2.2.3, we have:

$$\begin{aligned}
&\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^*\|_2^2\\
\leq&\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k} - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^{k} - \boldsymbol{\mu}^*\|_2^2\\
-&\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 - \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 - \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^k\|_2^2\\
+&\sum_{\mathrm{i}=1}^{N}\underbrace{2\rho(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\gamma}^{k+1})^T A_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k)}_{(a)_\mathrm{i}} + \sum_{\mathrm{j}=1}^{M}\sum_{\mathrm{i}=1}^{N}\underbrace{2\rho(\mu_\mathrm{j}^{k+1} - \nu_\mathrm{j}^{k+1})\Big[g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) - g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^k)\Big]}_{(b)_\mathrm{ji}}. \qquad (2.33)
\end{aligned}$$

Before we continue with the proof, we first show an extension of the Young's inequality[2] on vector products that will play a key role in the following proof. Given any two vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n$, we have that

$$\mathbf{z}_1^T \mathbf{z}_2 = \sum_{\mathrm{j}=1}^{n} z_{1\mathrm{j}} z_{2\mathrm{j}} = \sum_{\mathrm{j}=1}^{n} \Big(\frac{1}{\delta} z_{1\mathrm{j}}\Big)\Big(\delta z_{2\mathrm{j}}\Big) \leq \sum_{\mathrm{j}=1}^{n} \Big|\frac{1}{\delta} z_{1\mathrm{j}}\Big|\Big|\delta z_{2\mathrm{j}}\Big|,$$

where δ is a non-zero real number. Applying Young's inequality on each summation term with $p = q = 2$, we obtain that

$$\mathbf{z}_1^T \mathbf{z}_2 \leq \sum_{\mathrm{j}=1}^{n} \left[\frac{1}{2}\Big(\frac{1}{\delta} z_{1\mathrm{j}}\Big)^2 + \frac{1}{2}\Big(\delta z_{2\mathrm{j}}\Big)^2\right] = \frac{1}{2\delta^2}\|\mathbf{z}_1\|_2^2 + \frac{\delta^2}{2}\|\mathbf{z}_2\|_2^2. \tag{2.34}$$

Applying (2.34) on each term $(a)_\mathrm{i}$ yields

$$\begin{aligned}(a)_\mathrm{i} \leq & 2\rho\Big[\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|A_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k)\|_2^2\Big] \\ \leq & 2\rho\Big[\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|A_\mathrm{i}\|_2^2\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2\Big],\end{aligned} \tag{2.35}$$

and letting $\delta^2 = \frac{1}{\|A_\mathrm{i}\|_2}$ yields

$$\begin{aligned}(a)_\mathrm{i} \leq & \rho\|A_\mathrm{i}\|_2\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2\Big) \\ \leq & \rho A_{max}\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2\Big).\end{aligned} \tag{2.36}$$

Applying (2.34) on each term $(b)_\mathrm{ji}$ yields

$$\begin{aligned}(b)_\mathrm{ji} \leq & 2\rho\Big[\frac{1}{2\delta^2}\|\nu_\mathrm{j}^{k+1} - \mu_\mathrm{j}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) - g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^k)\|_2^2\Big] \\ \leq & 2\rho\Big[\frac{1}{2\delta^2}\|\nu_\mathrm{j}^{k+1} - \mu_\mathrm{j}^{k+1}\|_2^2 + \frac{\delta^2}{2}L_\mathrm{ji}^2\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2\Big],\end{aligned} \tag{2.37}$$

[2]Young's inequality states that if a and b are two non-negative real numbers, and p and q are real numbers greater than 1 such that $\frac{1}{p} + \frac{1}{q} = 1$, then $ab < \frac{a^p}{p} + \frac{b^q}{q}$.

and letting $\delta^2 = \frac{1}{L_{\text{ji}}}$ yields

$$\begin{aligned}(b)_{\text{ji}} \leq & \rho L_{\text{ji}}\Big(\|\nu_{\text{j}}^{k+1} - \mu_{\text{j}}^{k+1}\|_2^2 + \|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{k}\|_2^2\Big) \\ \leq & \rho L_{max}\Big(\|\nu_{\text{j}}^{k+1} - \mu_{\text{j}}^{k+1}\|_2^2 + \|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{k}\|_2^2\Big).\end{aligned} \tag{2.38}$$

Substituting (2.36) and (2.38) into (2.33) yields

$$\begin{aligned}&\sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^{*}\|_2^2 \\ \leq & \sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k} - \boldsymbol{\mu}^{*}\|_2^2 \\ & -(1 - \rho A_{max} - \rho M L_{max})\sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{k}\|_2^2 - (1 - \rho N A_{max})\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 \\ & -(1 - \rho N L_{max})\|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 - \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k}\|_2^2\end{aligned} \tag{2.39}$$

Since $0 < \rho \leq \min\left\{\frac{1-\epsilon}{A_{max}+ML_{max}}, \frac{1-\epsilon}{NA_{max}}, \frac{1-\epsilon}{NL_{max}}\right\}$, we have:

$$\begin{aligned}&\sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^{*}\|_2^2 \\ \leq & \sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k} - \boldsymbol{\mu}^{*}\|_2^2 \\ & -\epsilon\Big(\sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{k}\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 \\ & \quad + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k}\|_2^2\Big).\end{aligned} \tag{2.40}$$

It implies that for all $k \geq 0$:

$$\begin{aligned}0 \leq & \sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k+1} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^{*}\|_2^2 \\ \leq & \sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k} - \boldsymbol{\mu}^{*}\|_2^2 \\ \leq & \sum_{\text{i}=1}^{N}\|\mathbf{x}_{\text{i}}^{k-1} - \mathbf{x}_{\text{i}}^{*}\|_2^2 + \|\boldsymbol{\lambda}^{k-1} - \boldsymbol{\lambda}^{*}\|_2^2 + \|\boldsymbol{\mu}^{k-1} - \boldsymbol{\mu}^{*}\|_2^2\end{aligned}$$

$$\leq \cdots \leq \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^0 - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^0 - \boldsymbol{\mu}^*\|_2^2. \tag{2.41}$$

It further implies that the sequence $\left\{ \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^k - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^*\|_2^2 \right\}$ is monotonically decreasing and bounded below by 0; hence the sequence must be convergent to a limit, denoted by ξ:

$$\lim_{k\to+\infty} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^k - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^*\|_2^2 = \xi. \tag{2.42}$$

Taking the limit on both sides of (2.40) yields:

$$\begin{aligned}
&\lim_{k\to+\infty} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2 = 0, \\
&\lim_{k\to+\infty} \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 = 0, \qquad \lim_{k\to+\infty} \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 = 0, \\
&\lim_{k\to+\infty} \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 = 0, \qquad \lim_{k\to+\infty} \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k}\|_2^2 = 0.
\end{aligned} \tag{2.43}$$

Additionally, (2.42) also implies that $\{(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\}$ is a bounded sequence, and thus there exists a sub-sequence $\{(\mathbf{x}_1^{k_\mathrm{j}}, \ldots, \mathbf{x}_N^{k_\mathrm{j}}, \boldsymbol{\lambda}^{k_\mathrm{j}}, \boldsymbol{\mu}^{k_\mathrm{j}})\}$ that converges to a limit point $(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty)$. We next show that the limit point is indeed a saddle point and is also the unique limit point of $\{(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\}$. Applying Lemma 2.2.1 with $\hat{\mathbf{z}} = (\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1})$, $\bar{\mathbf{z}} = (\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k)$ and any $\mathbf{z} = (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \prod_{\mathrm{i}=1}^N \mathcal{X}_\mathrm{i}$, we have:

$$\begin{aligned}
&2\rho\Big[\mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1}) - \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\gamma}^{k+1}, \boldsymbol{\nu}^{k+1})\Big] \\
\leq &\sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^k - \mathbf{x}_\mathrm{i}\|_2^2 - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}\|_2^2 - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2 \\
\leq &\sum_{\mathrm{i}=1}^{N} \left(\|\mathbf{x}_\mathrm{i}^k - \mathbf{x}_\mathrm{i}^{k+1}\|^2 + \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}\|^2\right) - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}\|^2 - \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|^2 = 0 \\
&\forall (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \prod_{\mathrm{i}=1}^{N} \mathcal{X}_\mathrm{i}.
\end{aligned} \tag{2.44}$$

Taking the limits over an appropriate sub-sequence $\{k_{\mathrm{j}}\}$ on both sides and using (2.43), we have:

$$\mathcal{L}(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty) \leq \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty), \quad \forall (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \prod_{\mathrm{i}=1}^{N} \mathcal{X}_{\mathrm{i}}. \tag{2.45}$$

Similarly, applying Lemma 2.2.1 with $\widehat{\mathbf{z}} = (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$, $\bar{\mathbf{z}} = (\boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ and any $\mathbf{z} = (\boldsymbol{\lambda}, \boldsymbol{\mu} \in \mathbb{R}_+^{m_2})$, we have:

$$\begin{aligned}
&2\rho\left[\mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \mathcal{L}(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})\right] \\
\leq& \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}\|^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}\|^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|^2 \\
+& \|\boldsymbol{\mu}^k - \boldsymbol{\mu}\|^2 - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}\|^2 - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|^2 \\
\leq& \Big(\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k+1}\|^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}\|^2\Big) - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}\|^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|^2 \\
+& \Big(\|\boldsymbol{\mu}^k - \boldsymbol{\mu}^{k+1}\|^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}\|^2\Big) - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}\|^2 - \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|^2 = 0, \quad \forall \boldsymbol{\mu} \in \mathbb{R}_+^{m_2}.
\end{aligned} \tag{2.46}$$

Taking the limits over an appropriate sub-sequence $\{k_{\mathrm{j}}\}$ on both sides and using (2.43), we have:

$$\mathcal{L}(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}, \boldsymbol{\mu}) \leq \mathcal{L}(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty), \quad \forall \boldsymbol{\mu} \in \mathbb{R}_+^{m_2}. \tag{2.47}$$

Therefore, we show that $(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty)$ is indeed a saddle point of the Lagrangian function $\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \boldsymbol{\lambda}, \boldsymbol{\mu})$. Then (2.42) implies that

$$\lim_{k \to +\infty} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^k - \mathbf{x}_{\mathrm{i}}^\infty\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^\infty\|_2^2 + \|\boldsymbol{\mu}^k - \boldsymbol{\mu}^\infty\|_2^2 = \xi. \tag{2.48}$$

Since we have already argued (after Eq. (2.43)) that there exists a bounded sequence of $\{(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\}$ that converges to 0; that is, there exists $\{k_{\mathrm{j}}\}$ such that

$$\lim_{k_{\mathrm{j}} \to +\infty} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k_{\mathrm{j}}} - \mathbf{x}_{\mathrm{i}}^\infty\|_2^2 + \|\boldsymbol{\lambda}^{k_{\mathrm{j}}} - \boldsymbol{\lambda}^\infty\|_2^2 + \|\boldsymbol{\mu}^{k_{\mathrm{j}}} - \boldsymbol{\mu}^\infty\|_2^2 = 0,$$

which then implies that $\xi = 0$. Therefore, we show that $\{(\mathbf{x}_1^k, \ldots, \mathbf{x}_N^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\}$ converges globally to a saddle point $(\mathbf{x}_1^\infty, \ldots, \mathbf{x}_N^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\mu}^\infty)$.

2.6.3 Proof of Theorem 2.2.5

Letting

$$\begin{aligned}
\mathbf{u}_\mathrm{i}^k &= A_\mathrm{i}^T(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\gamma}^{k+1}) + \sum_{\mathrm{j}=1}^{m_2}(\mu_\mathrm{j}^{k+1} - \nu_\mathrm{j}^{k+1})\nabla_{\mathbf{x}_\mathrm{i}} g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) - \frac{1}{\rho}(\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k), \quad \forall \mathrm{i} = 1\dots N,\\
\mathbf{v}^k &= -\frac{1}{\rho}(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k),\\
\mathbf{w}^k &= -\frac{1}{\rho}(\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k),
\end{aligned} \tag{2.49}$$

we first show that $(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) \in S^{-1}(\mathbf{u}_1^k, \dots, \mathbf{u}_N^k, \mathbf{v}^k, \mathbf{w}^k)$. By the primal minimization step (2.4), we have, for all $\mathrm{i} = 1 \dots N$:

$$-\nabla_{\mathbf{x}_\mathrm{i}} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1}) - \underbrace{\left[A_\mathrm{i}^T\boldsymbol{\gamma}^{k+1} + \sum_{\mathrm{j}=1}^{M}\nu_\mathrm{j}^{k+1}\nabla_{\mathbf{x}_\mathrm{i}} g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) + \frac{1}{\rho}(\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k)\right]}_{\Delta_{u_\mathrm{i}}} \in \mathcal{N}_{\mathcal{X}_\mathrm{i}}(\mathbf{x}_\mathrm{i}^{k+1}), \tag{2.50}$$

where $\mathcal{N}_{\mathcal{X}_\mathrm{i}}(\mathbf{x}^{k+1}) \coloneqq \{\mathbf{y} \in \mathbb{R}^{n_1} | \mathbf{y}^T(\mathbf{x} - \mathbf{x}^{k+1}) \leq \mathbf{0}, \forall \mathbf{x} \in \mathcal{X}_\mathrm{i}\}$ denotes the normal cone to the set $\mathcal{X}_\mathrm{i}$ at the point $\mathbf{x}_\mathrm{i}^{k+1}$ for all $\mathrm{i} = 1, \dots, N$. Plugging

$$\Delta_{u_\mathrm{i}} = A_\mathrm{i}^T\boldsymbol{\lambda}^{k+1} + \sum_{\mathrm{j}=1}^{M}\mu_\mathrm{j}^{k+1}\nabla_{\mathbf{x}_\mathrm{i}} g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) - \mathbf{u}_\mathrm{i}^k$$

into the above expression, we have that, for all $\mathrm{i} = 1, \dots, N$:

$$-\nabla_{\mathbf{x}_\mathrm{i}} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1}) - A_\mathrm{i}^T\boldsymbol{\lambda}^{k+1} - \sum_{\mathrm{j}=1}^{M}\mu_\mathrm{j}^{k+1}\nabla_{\mathbf{x}_\mathrm{i}} g_\mathrm{ji}(\mathbf{x}_\mathrm{i}^{k+1}) + \mathbf{u}_\mathrm{i}^k \in \mathcal{N}_{\mathcal{X}_\mathrm{i}}(\mathbf{x}_\mathrm{i}^{k+1}), \tag{2.51}$$

which implies

$$\begin{aligned}
&(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1})\\
&\in \underset{(\mathbf{x}_1,\dots,\mathbf{x}_N)\in\prod_{\mathrm{i}=1}^N \mathcal{X}_\mathrm{i}}{\operatorname{argmin}} \mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_N, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) - \sum_{\mathrm{i}=1}^{N}\mathbf{x}_\mathrm{i}^T\mathbf{u}_\mathrm{i}^k + (\boldsymbol{\lambda}^{k+1})^T\mathbf{v}^k + (\boldsymbol{\mu}^{k+1})^T\mathbf{w}^k.
\end{aligned} \tag{2.52}$$

Similarly, by the interpretation of $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1})$ in Lemma 2.2.2, we have:

$$\begin{aligned} &\nabla_{\boldsymbol{\lambda}} \mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) + \underbrace{\left[-\frac{1}{\rho}(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k) \right]}_{\mathbf{v}^k} = \mathbf{0}, \\ &\nabla_{\boldsymbol{\mu}} \mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) + \underbrace{\left[-\frac{1}{\rho}(\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k) \right]}_{\mathbf{w}^k} \in \mathcal{N}_{\mathbb{R}_+^{m_2}}(\boldsymbol{\mu}^{k+1}), \end{aligned} \tag{2.53}$$

which imply

$$(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) \in \underset{\boldsymbol{\lambda} \in \mathbb{R}^m, \boldsymbol{\mu} \in \mathbb{R}^M}{\operatorname{argmax}} \mathcal{L}(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \sum_{\mathrm{i}=1}^{N} (\mathbf{x}_\mathrm{i}^{k+1})^T \mathbf{u}_\mathrm{i}^k + \boldsymbol{\lambda}^T \mathbf{v}^k + \boldsymbol{\mu}^T \mathbf{w}^k. \tag{2.54}$$

The first-order optimality conditions (2.52) and (2.54) together imply that

$$(\mathbf{x}_1^{k+1}, \dots, \mathbf{x}_N^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\mu}^{k+1}) \in S^{-1}(\mathbf{u}_1^k, \dots, \mathbf{u}_N^k, \mathbf{v}^k, \mathbf{w}^k).$$

By (2.43), we have $\lim_{k\to\infty}(\mathbf{u}_1^k, \dots, \mathbf{u}_N^k, \mathbf{v}^k, \mathbf{w}^k) \to \mathbf{0}$. Choose in integer $\bar{k}$ such that, for all $k \geq \bar{k}$, $\|(\mathbf{u}_1^k, \dots, \mathbf{u}_N^k, \mathbf{v}^k, \mathbf{w}^k)\|_2 \leq \tau$, then by Assumption 2.2.4, we have:

$$\begin{aligned} &\sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^*\|_2^2 \\ \leq& a^2 \left(\sum_{\mathrm{i}=1}^{N} \|\mathbf{u}_\mathrm{i}^k\|_2^2 + \|\mathbf{v}^k\|_2^2 + \|\mathbf{w}^k\|_2^2 \right) \\ \leq& a^2 \Bigg(N A_{max}^2 \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + N L_{max}^2 \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{1}{\rho^2} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2 \\ &\quad + \frac{1}{\rho^2} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{1}{\rho^2} \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\mu}^k\|_2^2 \Bigg) \\ \leq& a^2 \Bigg[\frac{1}{\rho^2} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^k\|_2^2 + N A_{max}^2 \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + N L_{max}^2 \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 \\ &\quad + \frac{1}{\rho^2} \left(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\gamma}^{k+1}\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 \right) \\ &\quad + \frac{1}{\rho^2} \left(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\mu}^k\|_2^2 \right) \Bigg] \end{aligned}$$

$$
\begin{aligned}
&\leq a^2\Bigg[\frac{1}{\rho^2}\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^{k}\|_2^2 \\
&\qquad+(NA_{max}^2+\frac{1}{\rho^2})\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2+(NL_{max}^2+\frac{1}{\rho^2})\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^{k+1}\|_2^2 \\
&\qquad+\frac{1}{\rho^2}\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2+\frac{1}{\rho^2}\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^{k}\|_2^2\Bigg] \\
&\leq a^2(N\alpha^2+\frac{1}{\rho^2})\Big(\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^{k}\|_2^2+\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^{k+1}\|_2^2 \\
&\qquad\qquad+\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\mu}^{k}\|_2^2\Big) \\
&\leq\frac{a^2(N\alpha^2+\frac{1}{\rho^2})}{\epsilon}\Bigg[\Big(\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k}-\mathbf{x}_\mathrm{i}^{*}\|_2^2+\|\boldsymbol{\lambda}^{k}-\boldsymbol{\lambda}^{*}\|_2^2+\|\boldsymbol{\mu}^{k}-\boldsymbol{\mu}^{*}\|_2^2\Big) \\
&\qquad-\Big(\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^{*}\|_2^2+\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^{*}\|_2^2+\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^{*}\|_2^2\Big)\Bigg].
\end{aligned}
\tag{2.55}
$$

The last inequality is due to (2.40), and $\alpha \coloneqq \max\{A_{max}, L_{max}\}$. We further derive

$$
\begin{aligned}
&\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k+1}-\mathbf{x}_\mathrm{i}^{*}\|_2^2+\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^{*}\|_2^2+\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\mu}^{*}\|_2^2 \\
\leq&\theta^2\Big(\sum_{\mathrm{i}=1}^{N}\|\mathbf{x}_\mathrm{i}^{k}-\mathbf{x}_\mathrm{i}^{*}\|_2^2+\|\boldsymbol{\lambda}^{k}-\boldsymbol{\lambda}^{*}\|_2^2+\|\boldsymbol{\mu}^{k}-\boldsymbol{\mu}^{*}\|_2^2\Big),
\end{aligned}
\tag{2.56}
$$

where $\theta=\sqrt{\frac{1}{1+\beta}}<1$ and $\beta=\frac{\epsilon}{a^2(N\alpha^2+\frac{1}{\rho^2})}>0$.

2.7 Proofs in Section 2.3.2

2.7.1 Proof of Theorem 2.3.2

For all $k \geq 0$, we can equivalently write the update steps in Algorithm 2 and Algorithm 3 as

$$
\mathbf{x}_\mathrm{i}^{k+1}=\begin{cases}\underset{\mathbf{x}_\mathrm{i}\in\mathcal{X}_\mathrm{i}}{\mathrm{argmin}} f_\mathrm{i}(\mathbf{x}_\mathrm{i})+(2\boldsymbol{\lambda}^{\hat{k}_\mathrm{i}+1}-\boldsymbol{\lambda}^{\hat{k}_\mathrm{i}})^T A_\mathrm{i}\mathbf{x}_\mathrm{i}+\frac{1}{2\rho}\|\mathbf{x}_\mathrm{i}-\mathbf{x}_\mathrm{i}^{\hat{k}_\mathrm{i}+1}\|^2, & \forall \mathrm{i}\in\mathcal{A}_k \\ \mathbf{x}_\mathrm{i}^{k}, & \forall\mathrm{i}\in\mathcal{A}_k^{\complement}\end{cases},
\tag{2.57}
$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho\Big(\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{b}\Big), \tag{2.58}$$

$$\hat{\gamma} = \boldsymbol{\lambda}^{k+1} + \rho\Big(\sum_{\mathrm{i}=1}^N A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{b}\Big) = 2\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k, \tag{2.59}$$

where $\hat{k}_\mathrm{i}$ is the last iteration when the main processor receives $\hat{\mathbf{x}}_\mathrm{i}$ from the worker processor $\mathrm{i} \in \mathcal{A}_k$ before iteration k. For each worker processor $\mathrm{i} \in \mathcal{A}_k^\complement$, we denote $\bar{k}_\mathrm{i} \in (k-\tau, k)$ as the last iteration when the main processor receives $\hat{\mathbf{x}}_\mathrm{i}$ from the worker processor i before iteration k, and further denote $\bar{\bar{k}}_\mathrm{i} \in [\bar{k}_\mathrm{i} - \tau, \bar{k}_\mathrm{i})$ as the last iteration when the main processor receives $\widehat{\mathbf{x}}_\mathrm{i}$ from the worker processor i before iteration $\bar{k}_\mathrm{i}$. We can rewrite the primal minimization step as

$$\mathbf{x}_\mathrm{i}^{k+1} = \begin{cases} \arg\min_{\mathbf{x}_\mathrm{i}} f_\mathrm{i}(\mathbf{x}_\mathrm{i}) + (2\boldsymbol{\lambda}^{\hat{k}_\mathrm{i}+1} - \boldsymbol{\lambda}^{\hat{k}_\mathrm{i}})^T A_\mathrm{i}\mathbf{x}_\mathrm{i} + \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i} - \mathbf{x}_\mathrm{i}^{\hat{k}_\mathrm{i}+1}\|^2, & \forall \mathrm{i} \in \mathcal{A}_k \\ \mathbf{x}_\mathrm{i}^{\bar{k}_\mathrm{i}+1} = \arg\min_{\mathbf{x}_\mathrm{i}} f_\mathrm{i}(\mathbf{x}_\mathrm{i}) + (2\boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}+1} - \boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}})^T A_\mathrm{i}\mathbf{x}_\mathrm{i} + \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i} - \mathbf{x}_\mathrm{i}^{\bar{\bar{k}}_\mathrm{i}+1}\|^2, & \forall \mathrm{i} \in \mathcal{A}_k^\complement \end{cases}. \tag{2.60}$$

At each iteration $k \geq 0$, for any $\mathrm{i} \in \mathcal{A}_k$, applying Lemma 2.3.1 with $\widehat{\mathbf{z}} = \mathbf{x}_\mathrm{i}^{k+1}$, $\bar{\mathbf{z}} = \mathbf{x}_\mathrm{i}^{\hat{k}_\mathrm{i}+1}$, and $\mathbf{z} = \mathbf{x}_\mathrm{i}^*$, we have:

$$\begin{aligned} & f_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1}) - f_\mathrm{i}(\mathbf{x}_\mathrm{i}^*) + \boldsymbol{\lambda}^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) + (\frac{\sigma_\mathrm{i}}{2} + \frac{1}{2\rho})\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \\ & + \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^{\hat{k}_\mathrm{i}+1}\|_2^2 - \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i}^{\hat{k}_\mathrm{i}+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \\ & + (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) + (2\boldsymbol{\lambda}^{\hat{k}_\mathrm{i}+1} - \boldsymbol{\lambda}^{\hat{k}_\mathrm{i}} - \boldsymbol{\lambda}^{k+1})^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \leq 0. \end{aligned} \tag{2.61}$$

At each iteration $k \geq 0$, for any $\mathrm{i} \in \mathcal{A}_k^\complement$, applying Lemma 2.3.1 with $\widehat{\mathbf{z}} = \mathbf{x}_\mathrm{i}^{k+1}$, $\bar{\mathbf{z}} = \mathbf{x}_\mathrm{i}^{\bar{\bar{k}}_\mathrm{i}+1}$, and $\mathbf{z} = \mathbf{x}_\mathrm{i}^*$, we have:

$$\begin{aligned} & f_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1}) - f_\mathrm{i}(\mathbf{x}_\mathrm{i}^*) + \boldsymbol{\lambda}^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) + (\frac{\sigma_\mathrm{i}}{2} + \frac{1}{2\rho})\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \\ & + \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^{\bar{\bar{k}}_\mathrm{i}+1}\|_2^2 - \frac{1}{2\rho}\|\mathbf{x}_\mathrm{i}^{\bar{\bar{k}}_\mathrm{i}+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \\ & + (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) + (2\boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}+1} - \boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}} - \boldsymbol{\lambda}^{k+1})^T\Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \leq 0. \end{aligned} \tag{2.62}$$

Summing (2.61) over all $\mathrm{i} \in \mathcal{A}_k$ and (2.62) over all $\mathrm{i} \in \mathcal{A}_k^{\complement}$ yields

$$\begin{aligned}
&\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^{k+1}) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^{*}) + \underbrace{\boldsymbol{\lambda}^T \sum_{\mathrm{i}=1}^{N} \Big(A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{*}\Big)}_{\text{(a)}} + \frac{\sigma_{min}}{2} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2 \\
&+ \underbrace{\frac{1}{2\rho} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2}_{\text{(b)}} \\
&+ \underbrace{\frac{1}{2\rho}\Big[\sum_{\mathrm{i}\in\mathcal{A}_k} \Big(\|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{\hat{k}_{\mathrm{i}}+1}\|_2^2 - \|\mathbf{x}_{\mathrm{i}}^{\hat{k}_{\mathrm{i}}+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2\Big) + \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} \Big(\|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{\bar{\bar{k}}_{\mathrm{i}}+1}\|_2^2 - \|\mathbf{x}_{\mathrm{i}}^{\bar{\bar{k}}_{\mathrm{i}}+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2\Big)\Big]}_{\text{(c)}} \\
&+ \underbrace{(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T \sum_{\mathrm{i}=1}^{N} \Big(A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{*}\Big)}_{\text{(d)}} \\
&+ \sum_{\mathrm{i}\in\mathcal{A}_k} (2\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}+1} - \boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}} - \boldsymbol{\lambda}^{k+1})^T \Big(A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{*}\Big) \\
&+ \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} (2\boldsymbol{\lambda}^{\bar{\bar{k}}_{\mathrm{i}}+1} - \boldsymbol{\lambda}^{\bar{\bar{k}}_{\mathrm{i}}} - \boldsymbol{\lambda}^{k+1})^T \Big(A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{*}\Big) \\
&\le 0.
\end{aligned} \tag{2.63}$$

The term (a) can be rewritten as:

$$(a) = \boldsymbol{\lambda}^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{*}\Big) = \boldsymbol{\lambda}^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{b}\Big). \tag{2.64}$$

The term (b) + (c) can be rewritten as:

$$\begin{aligned}
(b) + (c) =& \frac{1}{2\rho} \sum_{\mathrm{i}\in\mathcal{A}_k} \Big(\|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2 + \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{\hat{k}_{\mathrm{i}}+1}\|_2^2 - \|\mathbf{x}_{\mathrm{i}}^{\hat{k}_{\mathrm{i}}+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2\Big) \\
&+ \frac{1}{2\rho} \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} \Big(\|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2 + \|\mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{\bar{\bar{k}}_{\mathrm{i}}+1}\|_2^2 - \|\mathbf{x}_{\mathrm{i}}^{\bar{\bar{k}}_{\mathrm{i}}+1} - \mathbf{x}_{\mathrm{i}}^{*}\|_2^2\Big) \\
&\ge 0.
\end{aligned} \tag{2.65}$$

The term (d) can be rewritten as:

$$(d) = (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big) = \frac{1}{\rho} (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}). \tag{2.66}$$

We substitute (2.64), (2.65) and (2.66) into (2.63), and sum it over $k = 0 \dots K-1$. Taking the average yields

$$\begin{aligned}
&\frac{1}{K} \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^{k+1}) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^{*}) + \frac{1}{K} \boldsymbol{\lambda}^T \Big(\sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{b} \Big) \\
\leq & -\frac{\sigma_{min}}{2K} \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} \| \mathbf{x}_{\mathrm{i}}^{k+1} - \mathbf{x}_{\mathrm{i}}^{*} \|_2^2 \\
& - \frac{1}{\rho K} \underbrace{\sum_{k=0}^{K-1} (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda})^T (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k})}_{\text{(e)}} \\
& + \frac{1}{K} \underbrace{\sum_{k=0}^{K-1} \sum_{\mathrm{i} \in \mathcal{A}_k} (\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}} + \boldsymbol{\lambda}^{k+1} - 2\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}+1})^T \Big(A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big)}_{\text{(f)}} \\
& + \frac{1}{K} \underbrace{\sum_{k=0}^{K-1} \sum_{\mathrm{i} \in \mathcal{A}_k^{\complement}} (\boldsymbol{\lambda}^{\bar{\bar{k}}_{\mathrm{i}}} + \boldsymbol{\lambda}^{k+1} - 2\boldsymbol{\lambda}^{\bar{\bar{k}}_{\mathrm{i}}+1})^T \Big(A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big)}_{\text{(g)}}.
\end{aligned} \tag{2.67}$$

The term (e) in (2.67) can be rewritten as:

$$\begin{aligned}
\text{(e)} =& \frac{1}{2} \sum_{k=0}^{k-1} \Big(\| \boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda} \|_2^2 - \| \boldsymbol{\lambda} - \boldsymbol{\lambda}^{k} \|_2^2 + \| \boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k} \|_2^2 \Big) \\
=& \frac{1}{2} \| \boldsymbol{\lambda}^{K} - \boldsymbol{\lambda} \|_2^2 - \frac{1}{2} \| \boldsymbol{\lambda}^{0} - \boldsymbol{\lambda} \|_2^2 + \frac{1}{2} \sum_{k=0}^{K-1} \| \boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k} \|_2^2.
\end{aligned} \tag{2.68}$$

The term (f) in (2.67) can be bounded as:

$$\begin{aligned}
& \sum_{k=0}^{K-1} \sum_{\mathrm{i} \in \mathcal{A}_k} (\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}} + \boldsymbol{\lambda}^{k+1} - 2\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}+1})^T \Big(A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big) \\
=& \sum_{k=0}^{K-1} \sum_{\mathrm{i} \in \mathcal{A}_k} (\boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}} - \boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}+1})^T \Big(A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big) + \sum_{k=0}^{K-1} \sum_{\mathrm{i} \in \mathcal{A}_k} (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{\hat{k}_{\mathrm{i}}+1})^T \Big(A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{k+1} - A_{\mathrm{i}} \mathbf{x}_{\mathrm{i}}^{*} \Big)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k} \sum_{l=\hat{k}_\mathrm{i}}^{\hat{k}_\mathrm{i}} (\boldsymbol{\lambda}^l - \boldsymbol{\lambda}^{l+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&+ \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k} \sum_{l=\hat{k}_\mathrm{i}+1}^{k} (\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&\leq \sum_{\mathrm{i}\in\mathcal{A}_k} \sum_{k=0}^{K-1} \sum_{l=\hat{k}_\mathrm{i}}^{\hat{k}_\mathrm{i}} \left(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^l - \boldsymbol{\lambda}^{l+1}\|_2^2 + \frac{\delta^2\|A_\mathrm{i}\|_2^2}{2}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2\right) \\
&+ \sum_{\mathrm{i}\in\mathcal{A}_k} \sum_{k=0}^{K-1} \sum_{l=\hat{k}_\mathrm{i}+1}^{k} \left(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l}\|_2^2 + \frac{\delta^2\|A_\mathrm{i}\|_2^2}{2}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2\right) \\
&\leq \sum_{\mathrm{i}=1}^{N} \sum_{k=0}^{K-1} (\tau-1)\left(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k+1}\|_2^2 + \frac{\delta^2\|A_\mathrm{i}\|_2^2}{2}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2\right) \\
&+ \sum_{\mathrm{i}=1}^{N} \sum_{k=0}^{K-1} (\tau-1)\left(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \frac{\delta^2\|A_\mathrm{i}\|_2^2}{2}\|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2\right) \\
&\leq \frac{(\tau-1)N}{\delta^2} \sum_{k=0}^{K-1} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + (\tau-1)\delta^2 A_{\max}^2 \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2, \qquad (2.69)
\end{aligned}
$$

where the first inequality is obtained by (2.34), and the second inequality is due to the fact that the term $\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2$ does not appear more than $\tau - 1$ times for each iteration k.

Similarly, the term (g) in (2.67) can be bounded as:

$$
\begin{aligned}
&\sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} (\boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}} + \boldsymbol{\lambda}^{k+1} - 2\boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&= \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} (\boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}} - \boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) + \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} (\boldsymbol{\lambda}^{\bar{k}_\mathrm{i}+1} - \boldsymbol{\lambda}^{\bar{\bar{k}}_\mathrm{i}+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&+ \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} (\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{\bar{k}_\mathrm{i}+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&= \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} \sum_{l=\bar{\bar{k}}_\mathrm{i}}^{\bar{\bar{k}}_\mathrm{i}} (\boldsymbol{\lambda}^l - \boldsymbol{\lambda}^{l+1})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&+ \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} \sum_{l=\bar{\bar{k}}_\mathrm{i}+1}^{\bar{k}_\mathrm{i}} (\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big) \\
&+ \sum_{k=0}^{K-1} \sum_{\mathrm{i}\in\mathcal{A}_k^\complement} \sum_{l=\bar{k}_\mathrm{i}+1}^{k} (\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l})^T \Big(A_\mathrm{i}\mathbf{x}_\mathrm{i}^{k+1} - A_\mathrm{i}\mathbf{x}_\mathrm{i}^*\Big)
\end{aligned}
$$

$$
\begin{aligned}
&\leq \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} \sum_{k=0}^{K-1} \sum_{l=\bar{\bar{k}}_\mathrm{i}}^{\bar{\bar{k}}_\mathrm{i}} \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^l - \boldsymbol{\lambda}^{l+1}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&+ \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} \sum_{k=0}^{K-1} \sum_{l=\bar{\bar{k}}_\mathrm{i}+1}^{\bar{k}_\mathrm{i}} \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&+ \sum_{\mathrm{i}\in\mathcal{A}_k^{\complement}} \sum_{k=0}^{K-1} \sum_{l=\bar{k}_\mathrm{i}+1}^{k} \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^{l}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&\leq \sum_{\mathrm{i}=1}^{N} \sum_{k=0}^{K-1} (\tau - 1) \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&+ \sum_{\mathrm{i}=1}^{N} \sum_{k=0}^{K-1} (\tau - 1) \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&+ \sum_{\mathrm{i}=1}^{N} \sum_{k=0}^{K-1} (\tau - 1) \Big(\frac{1}{2\delta^2} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \frac{\delta^2 \|A_\mathrm{i}\|_2^2}{2} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2 \Big) \\
&\leq \frac{3(\tau-1)N}{2\delta^2} \sum_{k=0}^{K-1} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \frac{3(\tau-1)\delta^2}{2} A_{\max}^2 \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2. \qquad (2.70)
\end{aligned}
$$

By substituting (2.68), (2.69) and (2.70) into (2.67) and denoting

$$
\bar{\mathbf{x}}_\mathrm{i}^K = \frac{1}{K} \sum_{k=0}^{K-1} \mathbf{x}_\mathrm{i}^{k+1},
$$

for all $\mathrm{i} = 1 \ldots N$, we have:

$$
\begin{aligned}
&\sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\bar{\mathbf{x}}_\mathrm{i}^K) - \sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^*) + \boldsymbol{\lambda}^T \Big(\sum_{\mathrm{i}=1}^{N} A_\mathrm{i} \bar{\mathbf{x}}_\mathrm{i}^K - \mathbf{b} \Big) \\
\leq& \frac{1}{K} \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^{k+1}) - \sum_{\mathrm{i}=1}^{N} f_\mathrm{i}(\mathbf{x}_\mathrm{i}^*) + \frac{1}{K} \boldsymbol{\lambda}^T \sum_{k=0}^{K-1} \Big(\sum_{\mathrm{i}=1}^{N} A_\mathrm{i} \mathbf{x}_\mathrm{i}^{k+1} - \mathbf{b} \Big) \\
\leq& \frac{1}{2\rho K} \|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}\|_2^2 - \frac{1}{2\rho K} \|\boldsymbol{\lambda}^K - \boldsymbol{\lambda}\|_2^2 \\
+& \Big(-\frac{1}{2\rho K} + \frac{5(\tau-1)N}{2\delta^2 K}\Big) \sum_{k=0}^{K-1} \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 \\
+& \Big(-\frac{\sigma_{min}}{2K} + \frac{5(\tau-1)\delta^2 A_{\max}^2}{2K}\Big) \sum_{k=0}^{K-1} \sum_{\mathrm{i}=1}^{N} \|\mathbf{x}_\mathrm{i}^{k+1} - \mathbf{x}_\mathrm{i}^*\|_2^2, \qquad (2.71)
\end{aligned}
$$

where the first inequality is due to the convexity of f_{i} for all $\mathrm{i} = 1 \dots N$. By choosing $\delta^2 \leq \frac{\sigma_{min}}{5(\tau-1)A_{\max}^2}$ and $\rho \leq \frac{\delta^2}{5(\tau-1)N}$, which implies

$$\rho \leq \frac{\sigma_{min}}{25N(\tau-1)^2 A_{\max}^2},$$

we derive:

$$\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^K) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^*) + \boldsymbol{\lambda}^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}_{\mathrm{i}}^K - \mathbf{b} \Big) \leq \frac{1}{2\rho K} \|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}\|_2^2. \tag{2.72}$$

Let $\boldsymbol{\lambda} = \boldsymbol{\lambda}^* + \frac{\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b}}{\|\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b}\|_2}$, and note that by the duality theory, we have:

$$\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^K) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^*) + (\boldsymbol{\lambda}^*)^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}_{\mathrm{i}}^K - \mathbf{b} \Big) \geq 0. \tag{2.73}$$

Then, we further derive:

$$\begin{aligned}
&\Big\| \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b} \Big\|_2 \\
\leq &\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^K) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^*) + (\boldsymbol{\lambda}^*)^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}_{\mathrm{i}}^K - \mathbf{b} \Big) + \Big\| \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b} \Big\|_2 \\
\leq &\frac{1}{2\rho K} \left\| \boldsymbol{\lambda}^0 - \Big(\boldsymbol{\lambda}^* + \frac{\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b}}{\|\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b}\|_2} \Big) \right\|_2^2,
\end{aligned} \tag{2.74}$$

which implies

$$\Big\| \sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}^K - \mathbf{b} \Big\|_2 \leq \frac{1}{K} \Big[\frac{1}{2\rho} \max_{\|\boldsymbol{\gamma}\|_2 \leq 1} \|\boldsymbol{\lambda}^0 - \boldsymbol{\lambda}^* - \boldsymbol{\gamma}\|_2^2 \Big] \triangleq \frac{C_1}{K}.$$

On the other hand, let $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$, and note that:

$$\begin{aligned}
&\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^K) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^*) + (\boldsymbol{\lambda}^*)^T \Big(\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}_{\mathrm{i}}^K - \mathbf{b} \Big) \\
\geq &\Big| \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^K) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^*) \Big| - \|\boldsymbol{\lambda}^*\|_2 \cdot \|\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}} \bar{\mathbf{x}}_{\mathrm{i}}^K - \mathbf{b}\|_2.
\end{aligned} \tag{2.75}$$

Then, we have:

$$\Big|\sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\bar{\mathbf{x}}_{\mathrm{i}}^{K}) - \sum_{\mathrm{i}=1}^{N} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}^{*})\Big| \leq \|\boldsymbol{\lambda}^{*}\|_2 \cdot \|\sum_{\mathrm{i}=1}^{N} A_{\mathrm{i}}\bar{\mathbf{x}}_{\mathrm{i}}^{K} - \mathbf{b}\|_2 + \frac{1}{K}(\frac{1}{2\rho}\|\boldsymbol{\lambda}^{0} - \boldsymbol{\lambda}^{*}\|_2^2) \triangleq \frac{\delta_{\boldsymbol{\lambda}} C_1 + C_2}{K}, \quad (2.76)$$

where $\delta_{\boldsymbol{\lambda}} = \|\boldsymbol{\lambda}^{*}\|_2$ and $C_2 = \frac{1}{2\rho}\|\boldsymbol{\lambda}^{0} - \boldsymbol{\lambda}^{*}\|_2^2$.

3. A DISTRIBUTED ALGORITHM FOR LARGE-SCALE CONVEX QUADRATICALLY CONSTRAINED QUADRATIC PROGRAMS

3.1 Introduction

While in the last chapter, we propose distributed algorithms for solving block-separable convex optimization problems with coupling constraints, in this work, we focus on developing distributed algorithms for solving large-scale constrained convex optimization problems with non-separable objective function and nonlinear coupling constraints. More specifically, we consider the following constrained optimization problem:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^{n_1}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x} + \mathbf{q}_0^T \mathbf{x} + r_0 \\ \text{subject to} \quad & \frac{1}{2}\mathbf{x}^T P_{\mathrm{i}} \mathbf{x} + \mathbf{q}_{\mathrm{i}}^T \mathbf{x} + r_{\mathrm{i}} \le 0, \quad \mathrm{i} = 1, \ldots, m_1, \end{aligned} \tag{3.1}$$

where $P_{\mathrm{i}} \in \mathbb{R}^{n_1 \times n_1}$, $\mathbf{q}_{\mathrm{i}} \in \mathbb{R}^{n_1}$, and $r_{\mathrm{i}} \in \mathbb{R}$ for $\mathrm{i} = 0, 1, \ldots, m_1$ are all given. Such a problem is referred to as a quadratically constrained quadratic program (QCQP). (Note that linear constraints are included with $P_{\mathrm{i}} = \mathbf{0}$, a matrix of all 0's, for some i.) If additionally, $P_0, P_1, \ldots, P_{m_1}$ are all positive semidefinite (PSD) matrices, then the problem is convex. Convex QCQPs arise from a wide range of application areas, including multiple kernel learning [21], signal processing [22], radar applications [23], computer vision [24], and electric power system operation [25], to name a few. Small to medium-sized convex QCQPs can be solved efficiently by the well-established interior-point method (IPM) [26], which has polynomial running time for solving convex optimization problems. However, in order to write out the barrier function in the IPM for the feasible domain of a QCQP, decomposition of matrices $P_{\mathrm{i}} = F_{\mathrm{i}}^T F_{\mathrm{i}}$ for $\mathrm{i} = 1, \ldots, m_1$ is usually required [27], which may not be readily available through the input data. For example, in kernel-based learning applications, each quadratic constraint comprises a kernel matrix, whose components are directly defined by a kernel function: $K_{\mathrm{jj'}} = k(\mathbf{x}_{\mathrm{j}}, \mathbf{x}_{\mathrm{j'}})$. The operations to obtain a matrix decomposition, such as through Cholesky decomposition, typically have computational complexity of $O(n^3)$, which could become very costly as the size of the matrices grows. When the scale of the QCQPs increases dramatically due to huge amount of data, or when the data just cannot be all stored in a central location, a centralized algorithm, such as the IPM, may no longer be applicable. This directly motivates the proposed algorithm in this chapter, which facilitates distributed storage of data to achieve memory efficiency, does not require any matrix decomposition, and enables parallel computing even for QCQPs of non-separable constraints.

In addition to being a typical optimization problem, a convex QCQP is also a special instance of a second-order cone program (SOCP), which is in turn a special form of semi-definite program (SDP) [28]. When using commercial solvers, such as CPLEX, to solve a convex QCQP, it is usually transformed into an SOCP through preprocessing [29], and then a barrier-method-based optimizer is applied. To solve large-scale conic programs, [30] applies an operator splitting method (such as the well-known ADMM algorithm) to the homogeneous self-dual embedding, which is an equivalent convex feasibility problem involving finding a nonzero point in the intersection of a subspace and a cone. There are also ADMM-based distributed algorithms for solving large-scale SDPs proposed in [31], [32]; but they can only be applied to a class of decomposable SDPs with special graph representations (chordal graphs, for example). To translate a convex QCQP to either a standard SOCP or an SDP using the Schur Complement to rewrite each quadratic inequality as a linear matrix inequality (LMI), however, calls for matrix decomposition: $P_{\mathrm{i}} = F_{\mathrm{i}}^T F_{\mathrm{i}}$ for $\mathrm{i} = 1, \ldots, m_1$. As mentioned before, such operations can be very expensive for large-scale matrices. There is another ADMM-based distributed algorithm that decomposes a general QCQP with m constraints into m single-constrained QCQPs using a reformulated consensus optimization form [33]. However, even the size of the single-constrained QCQP can be very large in many applications, which may still need further decomposition, making the overall algorithm's efficiency in doubt. There is also a recent approach to transform quadratic constraints into linear constraints by sampling techniques and then to apply ADMM-based algorithms to solve the resulting large-scale quadratic programs (QPs) [34]. This approach is studied only for QCQPs with all matrices being positive definite (PD), and all the test problems shown in [34] are of a single constraint. How would the sampling approach perform with PSD matrices in the constraints or with multiple quadratic constraints is unknown.

To overcome the above-mentioned limitations of the existing algorithms, we propose a novel first-order distributed algorithm, which decomposes a convex QCQP by a method inspired by the idea of the PCPM algorithm [7]. The advantages of our algorithm include the following: (i) non-separable, quadratic functions can become naturally separable after introducing the so-called predictor and corrector variables for both primal and dual variables, which greatly facilities distributed computing; while ADMM-type algorithms cannot be directly applied to QCQPs without separable constraints; (ii) both the primal/dual predictor variables and corrector variables can be updated component-wise, making the method well-suited for massively parallel computing, and each n-by-n Hessian matrix can be stored column-wise in distributed computing units; (iii) no matrix decomposition or inversion is needed.

Convergence of our algorithm to an optimal solution will be shown, along with various numerical results. We first test the algorithm on solving standard QCQPs with randomly generated data sets of different scales, and then apply it to solve large-scale multiple kernel learning problems. Numerical experiments are conducted on a multi-node computer cluster through message passing interface (MPI), and multiple nodes are used to highlight the benefits of distributed implementation of our algorithm. Numerical results are compared with those obtained from the commercial solver CPLEX (version 12.8.0, using the barrier optimizer). The comparison will show that our algorithm can scale to very large problems at the cost of consuming more cheap iterations to reach a high accuracy. With a modest accuracy, our algorithm exhibits favorable scalability for solving large-scale QCQPs when CPLEX fails to provide a solution due to memory limit or other issues.

The remainder of the chapter is organized as follows. In Section 3.2, we briefly summarize the original PCPM algorithm and highlight the novel idea in our proposed algorithm. Section 3.3 provides convergence analyses of the algorithm, followed by discussions on how to implement the algorithm in a distributed framework in Section 3.4. Numerical performance of various testing problems is reported in Section 3.5. Finally, we conclude with some discussions in Section 3.6.

3.2 A Distributed Algorithm for Large-scale Convex QCQPs

Consider a convex QCQP problem in the following form:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{X},\ \mathbf{u}\in\mathbb{R}^{n_2}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}^T P_0\mathbf{x} + \mathbf{q}_0^T\mathbf{x} + \mathbf{c}_0^T\mathbf{u} + r_0 \\ \text{subject to} \quad & \frac{1}{2}\mathbf{x}^T P_\mathrm{i}\mathbf{x} + \mathbf{q}_\mathrm{i}^T\mathbf{x} + \mathbf{c}_\mathrm{i}^T\mathbf{u} + r_\mathrm{i} \le 0, \quad \mathrm{i}=1,\dots,m_1, \qquad (\lambda_\mathrm{i}) \\ & A\mathbf{x} + B\mathbf{u} = \mathbf{b}, \qquad (\boldsymbol{\gamma}) \end{aligned} \tag{3.2}$$

where $P_\mathrm{i} \in \mathbb{R}^{n_1\times n_1}$, $\mathbf{q}_\mathrm{i} \in \mathbb{R}^{n_1}$, $r_\mathrm{i} \in \mathbb{R}$ for $\mathrm{i} = 0, 1, \dots, m_1$, $A \in \mathbb{R}^{m_2\times n_1}$, $B \in \mathbb{R}^{m_2\times n_2}$ and $\mathbf{b} \in \mathbb{R}^{m_2}$ are all given. Note that we introduce a new variable $\mathbf{u} \in \mathbb{R}^{n_2}$ to explicitly write out the linear-only terms $\mathbf{c}_\mathrm{i}^T\mathbf{u}$ with coefficients $\mathbf{c}_\mathrm{i} \in \mathbb{R}^{n_2}$ for $\mathrm{i} = 0, 1, \dots, m_1$, and also write out a linear equality constraint $A\mathbf{x} + B\mathbf{u} = \mathbf{b}$ separately. While the generic set $\mathbb{X}$ can be any polyhedral set, we consider specifically the box constraints here; that is, $\mathbb{X} = \prod_{\mathrm{j}=1}^{n_1} \mathbb{X}_\mathrm{j} \subset \mathbb{R}^{n_1}$ of box constraint sets $\mathbb{X}_\mathrm{j} = \{x_\mathrm{j} \in \mathbb{R} | 0 \le x_\mathrm{j} \le \bar{X}_\mathrm{j}\}$ for $\mathrm{j} = 1, \dots, n_1$.

The specific QCQP formulation in (3.2) is not more general than the standard form (3.1). The reason that we write out a QCQP in this specific form is to emphasize the fact that when dealing with QCQPs with linear constraints (including box constraints), our algorithm does not require the problem to be reformulated into the standard form in (3.1). This can be convenient from implementation perspective, as several applications, including multiple kernel learning, naturally lead to a QCQP in the form of (3.2).

To avoid technical difficulties, we make the blanket assumption throughout this chapter that the Slater's constraint qualification (CQ) holds; consequently, a Lagrangian multiplier $(\boldsymbol{\lambda}, \boldsymbol{\gamma}) = (\lambda_1 \cdots \lambda_{m_1}, \gamma_1 \cdots \gamma_{m_2})^T$ always exists for any feasible point $(\mathbf{x}, \mathbf{u})$ of (3.2). To apply the PCPM algorithm to the QCQP in (3.2), at each iteration k, with a given primal-dual pair, $(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$, we start with a dual predictor update:

$$\begin{aligned} & \text{(dual predictor)}: \\ & \mu_\mathrm{i}^{k+1} = \Pi_{\mathbb{R}_+}\Big(\lambda_\mathrm{i}^k + \rho\Big[\frac{1}{2}(\mathbf{x}^k)^T P_\mathrm{i}\mathbf{x}^k + \mathbf{q}_\mathrm{i}^T\mathbf{x}^k + \mathbf{c}_\mathrm{i}^T\mathbf{u}^k + r_\mathrm{i}\Big]\Big), \quad \mathrm{i}=1,\dots,m_1, \\ & \nu_\mathrm{i}^{k+1} = \gamma_\mathrm{i}^k + \rho\Big[A\mathbf{x}^k + B\mathbf{u}^k - \mathbf{b}\Big]_\mathrm{i}, \quad \mathrm{i}=1,\dots,m_2, \end{aligned} \tag{3.3}$$

where $\Pi_{\mathbb{Z}}(\mathbf{z})$ denotes the projection of a vector $\mathbf{z} \in \mathbb{R}^n$ onto a set $\mathbb{Z} \subset \mathbb{R}^n$, and $\mathbb{R}_+$ refers to the set of all non-negative real numbers.

After the dual predictor update, we update the primal variables $(\mathbf{x}^{k+1}, \mathbf{u}^{k+1})$ by minimizing the augmented Lagrangian function $\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1})$ evaluated at the dual predictor variable $(\boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1})$, plus the proximal terms. The primal minimization step can be written as

$$\begin{aligned}\mathbf{x}^{k+1} = \underset{\mathbf{x}\in\mathbb{X}}{\operatorname{argmin}}\ & \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x} + \mathbf{q}_0^T \mathbf{x} + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big(\frac{1}{2}\mathbf{x}^T P_{\mathrm{i}} \mathbf{x} + \mathbf{q}_{\mathrm{i}}^T \mathbf{x}\Big) \\ & + (\boldsymbol{\nu}^{k+1})^T A\mathbf{x} + \frac{1}{2\rho}\|\mathbf{x} - \mathbf{x}^k\|_2^2, \end{aligned} \tag{3.4a}$$

$$\mathbf{u}^{k+1} = \underset{\mathbf{u}\in\mathbb{R}^{n_2}}{\operatorname{argmin}}\ \mathbf{c}_0^T \mathbf{u} + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1} \mathbf{c}_{\mathrm{i}}^T \mathbf{u} + (\boldsymbol{\nu}^{k+1})^T B\mathbf{u} + \frac{1}{2\rho}\|\mathbf{u} - \mathbf{u}^k\|_2^2. \tag{3.4b}$$

Introducing the dual predictors $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ allows parallel updating of the primal variables $\mathbf{x}$ and $\mathbf{u}$, exactly as in the general PCPM algorithm. However, the primal variable $\mathbf{x} = (x_1 \dots x_{\mathrm{j}} \dots x_{n_1})^T$ cannot be further decomposed into parallel updating of each component x_{j}, due to the coupling terms $\mathbf{x}^T P_{\mathrm{i}} \mathbf{x}$, $\mathrm{i} = 0, \dots, m_1$, unless all P_{i}'s are diagonal matrices. To realize parallel updating of x_{j}'s, we propose a simple idea to use $P_{\mathrm{i}}\mathbf{x}^k$ as a "predictor" for $P_{\mathrm{i}}\mathbf{x}$ in the optimization (3.4a).

To illustrate the idea, it may be easier to consider the first-order optimality conditions of (3.4a):

$$\frac{1}{\rho}(\mathbf{x}^k - \mathbf{x}^{k+1}) \in \underbrace{P_0\mathbf{x}^{k+1}}_{(\Delta_0)} + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big(\underbrace{P_{\mathrm{i}}\mathbf{x}^{k+1}}_{(\Delta_{\mathrm{i}})} + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\nu}^{k+1} + \mathcal{N}_{\mathbb{X}}(\mathbf{x}^{k+1}), \tag{3.5}$$

where $\mathcal{N}_{\mathbb{X}}(\mathbf{x}^{k+1})$ is the normal cone to the convex set $\mathbb{X} = \prod_{\mathrm{j}=1}^{n_1} \mathbb{X}_{\mathrm{j}}$ at the solution point $\mathbf{x}^{k+1}$. By approximating each (Δ_{i}) using the predictor $P_{\mathrm{i}}\mathbf{x}^k$, $\mathrm{i} = 0, 1, \dots, m_1$, the first-order optimality condition now becomes

$$\frac{1}{\rho}(\mathbf{x}^k - \mathbf{x}^{k+1}) \in P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\nu}^{k+1} + \mathcal{N}_{\mathbb{X}}(\mathbf{x}^{k+1}). \tag{3.6}$$

With (3.6), it is easy to see that $\mathbf{x}^{k+1}$ can be obtained through component-wise calculations. (Note that the normal cone of box constraints has explicit algebraic expressions and can also be decomposed component-wise with respect to $\mathbf{x}^{k+1}$.) Unfortunately, this simple idea would not work theoretically in the sense that convergence to an optimal solution cannot be established. This is mainly due to the difficulty to bound the error of $\|P_{\mathrm{i}}\mathbf{x}^{k+1} - P_{\mathrm{i}}\mathbf{x}^{k}\|$ along the iterations.

To overcome this hurdle, we propose a novel approach to split (3.6) into two steps by first introducing "primal predictor" variable $\mathbf{y}^{k+1}$ for the primal decision variable $\mathbf{x}^{k}$, followed by a corrector update:

$$\textbf{step 1 (predictor)} :$$

$$\frac{1}{\rho}(\mathbf{x}^k - \mathbf{y}^{k+1}) \in P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^k\Big(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\gamma}^k + \mathcal{N}_{\mathbb{X}}(\mathbf{y}^{k+1}); \tag{3.7a}$$

$$\textbf{step 2 (corrector)} :$$

$$\frac{1}{\rho}(\mathbf{x}^k - \mathbf{x}^{k+1}) \in P_0\mathbf{y}^{k+1} + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big(P_{\mathrm{i}}\mathbf{y}^{k+1} + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\nu}^{k+1} + \mathcal{N}_{\mathbb{X}}(\mathbf{x}^{k+1}). \tag{3.7b}$$

By focusing on box constraints for the generic set $\mathbb{X}_{\mathrm{j}}$, and using the notation $[\mathbf{z}]_{\mathrm{j}}$ to denote the j-th component of a vector $\mathbf{z}$, we can rewrite (3.7a) and (3.7b) component-wise as follows, for each $\mathrm{j} = 1, \ldots, n_1$:

$$\text{(primal predictor of } x_{\mathrm{j}}^k) :$$

$$y_{\mathrm{j}}^{k+1} := \Pi_{\mathbb{X}_{\mathrm{j}}}\left(x_{\mathrm{j}}^k - \rho\Big[P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^k\Big(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\gamma}^k\Big]_{\mathrm{j}}\right), \tag{3.8a}$$

$$\text{(primal corrector of } x_{\mathrm{j}}^k) :$$

$$x_{\mathrm{j}}^{k+1} = \Pi_{\mathbb{X}_{\mathrm{j}}}\left(x_{\mathrm{j}}^k - \rho\Big[P_0\mathbf{y}^{k+1} + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big(P_{\mathrm{i}}\mathbf{y}^{k+1} + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\nu}^{k+1}\Big]_{\mathrm{j}}\right), \tag{3.8b}$$

where the projection onto the box constraint set $\mathbb{X}$ can be expressed as:

$$\Pi_{\mathbb{X}_{\mathrm{j}}}(x_{\mathrm{j}}) :\, = \begin{cases} 0, & \text{if } x_{\mathrm{j}} < 0; \\ x_{\mathrm{j}}, & \text{if } 0 \le x_{\mathrm{j}} \le \bar{X}_{\mathrm{j}}; \\ \bar{X}_{\mathrm{j}}, & \text{if } x_{\mathrm{j}} > \bar{X}_{\mathrm{j}}. \end{cases} \tag{3.9}$$

With (3.8a) and (3.8b), in addition to the apparent benefits of updating the variables component-wise, which will allow massively parallel computing, the multiplications of $P_{\mathrm{i}}\mathbf{x}^k$ and $P_{\mathrm{i}}\mathbf{y}^{k+1}$, $\mathrm{i} = 0, \ldots, m_1$ in (3.8a) and (3.8b) do not need to be carried out completely; only the j-th column of each matrix P_{i} is needed to complete the updates for y_{j}^{k+1} and x_{j}^{k+1}. Such an observation will allow distributed storage of the potentially huge-sized matrices. More detailed discussions of this point are provided in Section 3.4.1.

The update of the other primal variable, $\mathbf{u}^{k+1}$, can be performed in a similar fashion, which is to split into two steps by first introducing a predictor variable $\mathbf{v}^{k+1}$ for $\mathbf{u}^k$, followed by a corrector update:

$$
\begin{aligned}
&\text{(primal predictor of } u_{\mathrm{j}}^k\text{) :}\\
&v_{\mathrm{j}}^{k+1} := u_{\mathrm{j}}^k - \rho\Big[\mathbf{c}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^k \mathbf{c}_{\mathrm{i}} + B^T \boldsymbol{\gamma}^k\Big]_{\mathrm{j}}, \quad \mathrm{j} = 1, \ldots, n_2,
\end{aligned} \tag{3.10a}
$$

$$
\begin{aligned}
&\text{(primal corrector of } u_{\mathrm{j}}^k\text{) :}\\
&u_{\mathrm{j}}^{k+1} = u_{\mathrm{j}}^k - \rho\Big[\mathbf{c}_0 + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1} \mathbf{c}_{\mathrm{i}} + B^T \boldsymbol{\nu}^{k+1}\Big]_{\mathrm{j}}, \quad \mathrm{j} = 1, \ldots, n_2.
\end{aligned} \tag{3.10b}
$$

A dual corrector update is then performed for each Lagrangian multiplier $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})$:

$$
\begin{aligned}
&\text{(dual corrector) :}\\
&\lambda_{\mathrm{i}}^{k+1} = \Pi_{\mathbb{R}_+}\Big(\lambda_{\mathrm{i}}^k + \rho\Big[\frac{1}{2}(\mathbf{y}^{k+1})^T P_{\mathrm{i}} \mathbf{y}^{k+1} + \mathbf{q}_{\mathrm{i}}^T \mathbf{y}^{k+1} + \mathbf{c}_{\mathrm{i}}^T \mathbf{v}^{k+1} + r_{\mathrm{i}}\Big]\Big),\\
&\qquad\qquad \mathrm{i} = 1, \ldots, m_1,\\
&\gamma_{\mathrm{i}}^{k+1} = \gamma_{\mathrm{i}}^k + \rho\Big[A\mathbf{y}^{k+1} + B\mathbf{v}^{k+1} - \mathbf{b}\Big]_{\mathrm{i}}, \quad \mathrm{i} = 1, \ldots, m_2.
\end{aligned} \tag{3.11}
$$

The overall structure of the proposed algorithm, which we name it PC^{2}PM, to reflect the fact that two sets of predictors and correctors are utilized, is presented in Algorithm 4 below.

Note that the starting point of the PC^{2}PD algorithm can be arbitrary, and is not required to be feasible. To establish convergence of the algorithm, the specific rules to update the step-size ρ are crucial, which is the main focus of the next section. The implementation details,

Algorithm 4 PC^{2}PM

```
1: Initialization choose an arbitrary starting point $(\mathbf{x}^0, \mathbf{u}^0, \boldsymbol{\lambda}^0, \boldsymbol{\gamma}^0)$.
2: $k \leftarrow 0$.
3: while termination conditions are not met do
4:     (Adaptive step-size)
        update the step-size $\rho^{k+1}$;
5:     (Predictor update)
        update $(\boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1})$, $\mathbf{y}^{k+1}$, and $\mathbf{v}^{k+1}$ according to (3.3), (3.8a) and (3.10a);
6:     (Corrector update)
        update $\mathbf{x}^{k+1}$, $\mathbf{u}^{k+1}$, and $(\boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})$ according to (3.8b), (3.10b) and (3.11);
7:     $k \leftarrow k + 1$
8: return $(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$.
```

including distributed data storage, parallel computing through Message Passing Interface (MPI), and termination conditions, are provided in Section 3.4.

3.3 Convergence Analysis

In this section, we establish sufficient conditions for the PC^{2}PM algorithm to converge to an optimal solution from any starting point. First, we make a standard assumption on (3.2) about the existence of a saddle point.

Assumption 3.3.1 (Existence of a Saddle Point)**.** For the Lagrangian function of (3.2):

$$\begin{aligned}\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) :=& \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x} + \mathbf{q}_0^T \mathbf{x} + \mathbf{c}_0^T \mathbf{u} + r_0 \\ &+ \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}} \Big(\frac{1}{2}\mathbf{x}^T P_{\mathrm{i}} \mathbf{x} + \mathbf{q}_{\mathrm{i}}^T \mathbf{x} + \mathbf{c}_{\mathrm{i}}^T \mathbf{u} + r_{\mathrm{i}} \Big) + \boldsymbol{\gamma}^T (A\mathbf{x} + B\mathbf{u} - \mathbf{b}),\end{aligned} \tag{3.12}$$

we assume that a saddle point $(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$ exists; that is, for any $\mathbf{x} \in \mathbb{X}$, $\mathbf{u} \in \mathbb{R}^{n_2}$, $\boldsymbol{\lambda} \in \mathbb{R}_+^{m_1}$ and $\boldsymbol{\gamma} \in \mathbb{R}^{m_2}$,

$$\mathcal{L}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}, \boldsymbol{\gamma}) \le \mathcal{L}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*) \le \mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*). \tag{3.13}$$

Note that coupled with the blanket assumption that Slater's CQ holds for the QCQP (3.2), the above assumption is equivalent to say that an optimal solution of (3.2) is assumed to exist.

Next, we derive some essential lemmas for constructing the main convergence proof. For the ease of presenting the next two lemmas, we first introduce a notation for the linear approximation of the Lagrangian function (3.12).

Definition 3.3.1. With a given tuple $(\mathbf{x}',\ \boldsymbol{\lambda}',\ \boldsymbol{\gamma}') \in \mathbb{X} \times \mathbb{R}_+^{m_1} \times \mathbb{R}^{m_2}$, we define the following function $\mathcal{R} : \mathbb{X} \times \mathbb{R}^{n_2} \to \mathbb{R}$ as a linear approximation of the Lagrangian function $\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \boldsymbol{\gamma})$ evaluated at $(\mathbf{x}', \boldsymbol{\lambda}', \boldsymbol{\gamma}')$.

$$\begin{aligned}\mathcal{R}(\mathbf{x}, \mathbf{u}; \mathbf{x}', \boldsymbol{\lambda}', \boldsymbol{\gamma}') \coloneqq{}& (P_0\mathbf{x}' + \mathbf{q}_0)^T\mathbf{x} + \mathbf{c}_0^T\mathbf{u} + r_0 \\ &+ \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}'\Big[(P_{\mathrm{i}}\mathbf{x}' + \mathbf{q}_{\mathrm{i}})^T\mathbf{x} + \mathbf{c}_{\mathrm{i}}^T\mathbf{u} + r_{\mathrm{i}}\Big] + (\boldsymbol{\gamma}')^T(A\mathbf{x} + B\mathbf{u} - \mathbf{b}),\end{aligned} \tag{3.14}$$

for any $\mathbf{x} \in \mathbb{X}$ and $\mathbf{u} \in \mathbb{R}^{n_2}$.

Lemma 3.3.1. The update steps (3.3), (3.8a), (3.8a), (3.10a), (3.10b) and (3.11) are equivalent to obtaining proximal minimization points as follows:

$$\begin{aligned}(\boldsymbol{\mu}^{k+1},\ \boldsymbol{\nu}^{k+1}) = \underset{\boldsymbol{\lambda}\in\mathbb{R}_+^{m_1},\ \boldsymbol{\gamma}\in\mathbb{R}^{m_2}}{\operatorname{argmin}} &- \mathcal{L}(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}, \boldsymbol{\gamma}) \\ &+ \frac{1}{2\rho^{k+1}}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^k\|_2^2 + \frac{1}{2\rho^{k+1}}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^k\|_2^2;\end{aligned} \tag{3.15a}$$

$$\begin{aligned}(\mathbf{y}^{k+1},\ \mathbf{v}^{k+1}) = \underset{\mathbf{x}\in\mathbb{X},\ \mathbf{u}\in\mathbb{R}^{n_2}}{\operatorname{argmin}} &\ \mathcal{R}(\mathbf{x}, \mathbf{u}; \mathbf{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k) \\ &+ \frac{1}{2\rho^{k+1}}\|\mathbf{x} - \mathbf{x}^k\|_2^2 + \frac{1}{2\rho^{k+1}}\|\mathbf{u} - \mathbf{u}^k\|_2^2;\end{aligned} \tag{3.15b}$$

$$\begin{aligned}(\mathbf{x}^{k+1},\ \mathbf{u}^{k+1}) = \underset{\mathbf{x}\in\mathbb{X},\ \mathbf{u}\in\mathbb{R}^{n_2}}{\operatorname{argmin}} &\ \mathcal{R}(\mathbf{x}, \mathbf{u}; \mathbf{y}^{k+1}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1}) \\ &+ \frac{1}{2\rho^{k+1}}\|\mathbf{x} - \mathbf{x}^k\|_2^2 + \frac{1}{2\rho^{k+1}}\|\mathbf{u} - \mathbf{u}^k\|_2^2;\end{aligned} \tag{3.15c}$$

$$\begin{aligned}(\boldsymbol{\lambda}^{k+1},\ \boldsymbol{\gamma}^{k+1}) = \underset{\boldsymbol{\lambda}\in\mathbb{R}_+^{m_1},\ \boldsymbol{\gamma}\in\mathbb{R}^{m_2}}{\operatorname{argmin}} &- \mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) \\ &+ \frac{1}{2\rho^{k+1}}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^k\|_2^2 + \frac{1}{2\rho^{k+1}}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^k\|_2^2.\end{aligned} \tag{3.15d}$$

Since all the four optimization in (3.15a) – (3.15d) are convex optimization problems with linear constraints, the proof follows directly from the first-order optimality conditions of each of the optimization problems, and hence is omitted.

Lemma 3.3.2. At a saddle point $(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$ of the QCQP (3.2), the following inequality holds for any $\mathbf{x} \in \mathbb{X}$, $\mathbf{u} \in \mathbb{R}^{n_2}$, $\boldsymbol{\lambda} \in \mathbb{R}^{m_1}_+$ and $\boldsymbol{\gamma} \in \mathbb{R}^{m_2}$:

$$\begin{aligned}&\mathcal{R}(\mathbf{x}^*, \mathbf{u}^*; \mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) - \mathcal{R}(\mathbf{x}, \mathbf{u}; \mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\gamma})\\ \leq &\sum_{\mathrm{i}=1}^{m_1} (\lambda_\mathrm{i}^* - \lambda_\mathrm{i})\Big(\frac{1}{2}\mathbf{x}^T P_\mathrm{i}\mathbf{x} + \mathbf{q}_\mathrm{i}^T\mathbf{x} + \mathbf{c}_\mathrm{i}^T\mathbf{u} + r_\mathrm{i}\Big) + (\boldsymbol{\gamma}^* - \boldsymbol{\gamma})^T(A\mathbf{x} + B\mathbf{u} - \mathbf{b}).\end{aligned} \tag{3.16}$$

Proof. For any $\mathbf{x} \in \mathbb{X}$, $\mathbf{u} \in \mathbb{R}^{n_2}$, $\boldsymbol{\lambda} \in \mathbb{R}^{m_1}_+$ and $\boldsymbol{\gamma} \in \mathbb{R}^{m_2}$, we have that $\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*) \geq \mathcal{L}(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}, \boldsymbol{\gamma})$ by the saddle point inequality (3.13). We also have the inequality $\frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T P_0(\mathbf{x} - \mathbf{x}^*) + \sum_{\mathrm{i}=1}^{m_1} \lambda_\mathrm{i}\Big[\frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T P_\mathrm{i}(\mathbf{x} - \mathbf{x}^*)\Big] \geq 0$ due to the positive semi-definiteness of each matrix $P_0, P_1, \ldots, P_{m_1}$. Adding the two inequalities together completes the proof. □

We next establish fundamental estimates of the distance between the solution point $(\mathbf{x}^{k+1}, \mathbf{u}^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})$ at each iteration k and the saddle point $(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$.

Proposition 3.3.3. Let $(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$ be a saddle point of the QCQP (3.2). For all $k \geq 0$, we have that

$$\begin{aligned}&\|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_2^2\\ \leq\ &\|\mathbf{x}^k - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^k - \mathbf{u}^*\|_2^2\\ -\ &\Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big)\\ +\ &2\rho^{k+1}\Bigg\{(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_0(\mathbf{y}^{k+1} - \mathbf{x}^k)\\ &\qquad + \sum_{\mathrm{i}=1}^{m_1} \mu_\mathrm{i}^{k+1}(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_\mathrm{i}(\mathbf{y}^{k+1} - \mathbf{x}^k)\\ &\qquad + \sum_{\mathrm{i}=1}^{m_1} (\lambda_\mathrm{i}^* - \mu_\mathrm{i}^{k+1})\left[\frac{1}{2}(\mathbf{y}^{k+1})^T P_\mathrm{i}\mathbf{y}^{k+1} + \mathbf{q}_\mathrm{i}^T\mathbf{y}^{k+1} + \mathbf{c}_\mathrm{i}^T\mathbf{v}^{k+1} + r_\mathrm{i}\right]\\ &\qquad + (\boldsymbol{\gamma}^* - \boldsymbol{\nu}^{k+1})^T(A\mathbf{y}^{k+1} + B\mathbf{v}^{k+1} - \mathbf{b})\end{aligned}$$

$$
\begin{aligned}
&+\sum_{\mathrm{i}=1}^{m_1}(\mu_{\mathrm{i}}^{k+1}-\lambda_{\mathrm{i}}^{k})\left[(P_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}})^T(\mathbf{y}^{k+1}-\mathbf{x}^{k+1})+\mathbf{c}_{\mathrm{i}}^T(\mathbf{v}^{k+1}-\mathbf{u}^{k+1})\right]\\
&+(\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k})^T\left[A(\mathbf{y}^{k+1}-\mathbf{x}^{k+1})+B(\mathbf{v}^{k+1}-\mathbf{u}^{k+1})\right]\Bigg\},
\end{aligned}
\tag{3.17}
$$

and

$$
\begin{aligned}
&\|\boldsymbol{\lambda}^{k+1}-\boldsymbol{\lambda}^*\|_2^2+\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\gamma}^*\|_2^2\\
\le{}&\|\boldsymbol{\lambda}^{k}-\boldsymbol{\lambda}^*\|_2^2+\|\boldsymbol{\gamma}^{k}-\boldsymbol{\gamma}^*\|_2^2\\
&-\left(\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1}\|_2^2+\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2\right)\\
&+2\rho^{k+1}\Bigg\{\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^{k+1}-\lambda_{\mathrm{i}}^{*})\Big[\frac{1}{2}(\mathbf{y}^{k+1})^TP_{\mathrm{i}}\mathbf{y}^{k+1}+\mathbf{q}_{\mathrm{i}}^T\mathbf{y}^{k+1}+\mathbf{c}_{\mathrm{i}}^T\mathbf{v}^{k+1}+r_{\mathrm{i}}\Big]\\
&\qquad+(\boldsymbol{\gamma}^{k+1}-\boldsymbol{\gamma}^*)^T(A\mathbf{y}^{k+1}+B\mathbf{v}^{k+1}-\mathbf{b})\\
&\qquad+\sum_{\mathrm{i}=1}^{m_1}(\mu_{\mathrm{i}}^{k+1}-\lambda_{\mathrm{i}}^{k+1})\Big[\frac{1}{2}(\mathbf{x}^{k})^TP_{\mathrm{i}}\mathbf{x}^{k}+\mathbf{q}_{\mathrm{i}}^T\mathbf{x}^{k}+\mathbf{c}_{\mathrm{i}}^T\mathbf{u}^{k}+r_{\mathrm{i}}\Big]\\
&\qquad+(\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1})^T(A\mathbf{x}^{k}+B\mathbf{u}^{k}-\mathbf{b})\Bigg\}.
\end{aligned}
\tag{3.18}
$$

Proof. The details of the proof are provided in Appendix 3.7. □

Now we are ready to present the main convergence result.

Theorem 3.3.4 (Global Convergence)**.** Assume that the Slater's CQ and Assumption 3.3.1 hold. With a given scalar $0 \le \epsilon_0 < 1$, and a series of positive scalars $\epsilon_s > 0, s = 1, \ldots, 8$ that satisfy $\sum_{s=1}^{8}\epsilon_s \le 1-\epsilon_0$, we define the following function $\rho : \mathbb{X}\times\mathbb{R}^{n_2}\times\mathbb{R}_+^{m_1}\times\mathbb{R}^{m_2} \to (0,+\infty)$ to update the adaptive step size ρ^{k+1} in Algorithm 4 at each iteration k:

$$
\begin{aligned}
\rho^{k+1} &= \rho(\mathbf{x}^k,\mathbf{u}^k,\boldsymbol{\lambda}^k,\boldsymbol{\gamma}^k)\\
&:= \min\left\{\rho_1,\rho_2(\mathbf{x}^k,\mathbf{u}^k,\boldsymbol{\lambda}^k),\rho_3(\mathbf{x}^k,\boldsymbol{\lambda}^k,\boldsymbol{\gamma}^k),\rho_4,\rho_5(\mathbf{x}^k),\rho_6,\rho_7,\rho_8\right\},
\end{aligned}
\tag{3.19}
$$

where

(i) $\rho_1 = \begin{cases} \dfrac{\epsilon_1}{\|P_0\|_F}, & \text{if } \|P_0\|_F \neq 0 \\ \epsilon_1, & \text{if } \|P_0\|_F = 0, \end{cases}$ with $\|\cdot\|_F$ representing the Frobenius norm of a matrix;

(ii) $\rho_2(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k) = \min_{\mathrm{i}}\{\rho_{2\mathrm{i}}(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k)\}$, where

$$\rho_{2\mathrm{i}}(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k) \coloneqq \begin{cases} \dfrac{-b_{\mathrm{i}} + \sqrt{b_{\mathrm{i}}^2 + 4a_{\mathrm{i}}c_{\mathrm{i}}}}{2a_{\mathrm{i}}}, & \text{if } a_{\mathrm{i}} > 0 \\ \dfrac{c_{\mathrm{i}}}{b_{\mathrm{i}}}, & \text{if } a_{\mathrm{i}} = 0, b_{\mathrm{i}} > 0 \\ M, & \text{if } a_{\mathrm{i}} = 0, b_{\mathrm{i}} = 0, \end{cases}$$

for all $\mathrm{i} = 1, \ldots, m_1$, with $a_{\mathrm{i}} = |\frac{1}{2}(\mathbf{x}^k)^T P_{\mathrm{i}} \mathbf{x}^k + \mathbf{q}_{\mathrm{i}}^T \mathbf{x}^k + \mathbf{c}_{\mathrm{i}}^T \mathbf{u}^k + r_{\mathrm{i}}| \geq 0$, and $b_{\mathrm{i}} = \lambda_{\mathrm{i}}^k \geq 0$. For c_{i}, if $\|P_{\mathrm{i}}\|_F \neq 0$, $c_{\mathrm{i}} = \frac{\epsilon_2}{m_1\|P_{\mathrm{i}}\|_F} > 0$; otherwise $c_{\mathrm{i}} = \frac{\epsilon_2}{m_1} > 0$. The constant $M > 0$ is a given scalar;

(iii) $\rho_3(\mathbf{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k) =$

$$\begin{cases} \min\{2\epsilon_3, \dfrac{-b + \sqrt{b^2 + 4ac}}{2a}\}, & \text{if } a > 0 \\ \min\{2\epsilon_3, \dfrac{c}{b}\}, & \text{if } a = 0, b > 0 \\ 2\epsilon_3, & \text{if } a = 0, b = 0, \end{cases}$$

where $a = \|P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^k(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}) + A^T\boldsymbol{\gamma}^k\|_2 \geq 0$, $b = 2\|\mathbf{x}^k\|_2 \geq 0$ and $c = \frac{2\epsilon_3}{\|P\|_F} > 0$ with $P \in \mathbb{R}^{m_1 n_1 \times n_1}$ denoting the stacked matrix $\begin{pmatrix} P_1 \\ \vdots \\ P_{m_1} \end{pmatrix}$;

(iv) $\rho_4 = \begin{cases} \dfrac{\epsilon_4}{\|Q\|_F}, & \text{if } \|Q\|_F \neq 0 \\ \epsilon_4, & \text{if } \|Q\|_F = 0 \end{cases}$, where $Q \in \mathbb{R}^{m_1 \times n_1}$ denotes matrix $\begin{pmatrix} \mathbf{q}_1^T \\ \vdots \\ \mathbf{q}_{m_1}^T \end{pmatrix}$, with the $\mathbf{q}_{\mathrm{i}}$'s being the vectors in the linear terms of $\mathbf{x}$ in the QCQP (3.2);

(v) $\rho_5(\mathbf{x}^k) = \begin{cases} \dfrac{\epsilon_5}{\|\mathbf{x}^k\|_2\|P\|_F}, & \text{if } \|\mathbf{x}^k\|_2 \neq 0 \\ \epsilon_5, & \text{if } \|\mathbf{x}^k\|_2 = 0 \end{cases}$;

(vi) $\rho_6 = \begin{cases} \dfrac{\epsilon_6}{\|C\|_F}, & \text{if } \|C\|_F \neq 0 \\ \epsilon_6, & \text{if } \|C\|_F = 0 \end{cases}$, where $C \in \mathbb{R}^{m_2 \times n_2}$ denotes matrix $\begin{pmatrix} \mathbf{c}_1^T \\ \vdots \\ \mathbf{c}_{m_2}^T \end{pmatrix}$, with the $\mathbf{c}_{\mathrm{j}}$'s being the vectors in the linear terms of $\mathbf{u}$ in the QCQP (3.2);

(vii) $\rho_7 = \begin{cases} \dfrac{\epsilon_7}{\|A\|_F}, & \text{if } \|A\|_F \neq 0 \\ \epsilon_7, & \text{if } \|A\|_F = 0 \end{cases}$, where A is the matrix in the linear constraint $A\mathbf{x}+B\mathbf{u} = \mathbf{b}$ in (3.2);

(viii) $\rho_8 = \begin{cases} \dfrac{\epsilon_8}{\|B\|_F}, & \text{if } \|B\|_F \neq 0 \\ \epsilon_8, & \text{if } \|B\|_F = 0 \end{cases}$, where B is the matrix in the linear constraint $A\mathbf{x} + B\mathbf{u} = \mathbf{b}$ in (3.2).

Let $\{(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)\}$ be the sequence generated by Algorithm 4, with an arbitrary starting point $(\mathbf{x}^0, \mathbf{u}^0, \boldsymbol{\lambda}^0, \boldsymbol{\gamma}^0) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}$; then the sequence converges to a saddle point $(\mathbf{x}^*, \mathbf{u}^*, \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$ of the QCQP (3.2).

Proof. Please see Appendix 3.7 for details. □

While the rules to update the step size ρ^{k+1} may appear to be very cumbersome, the calculations are actually quite straightforward. Since the values of Frobenius norm of all matrices can be obtained in advance, the values of ρ_1, ρ_4, ρ_6, ρ_7 and ρ_8 are pre-determined. Given a current solution $(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$, ρ_2, ρ_3 and ρ_5 can also be easily calculated. The minimum of all the ρ_s's then determines the value of the adaptive step size ρ^{k+1}.

Another point we want to emphasize is that the convergence result is quite strong, in the sense that the entire sequence, not just a subsequence, from the algorithm can be shown to converge to an optimization solution, with an arbitrary starting point. Such a result can help alleviate a strong assumption we made, which is to assume the feasibility of a convex QCQP. While detecting if a (convex) QCQP is feasible or not can be as difficult as solving the problem itself, from a practical perspective, our algorithm can just be blindly applied to a QCQP. If the iterations appear to be diverging, then because of the whole sequence convergence result in the above theorem, it likely indicates that the QCQP is infeasible.

3.4 Implementation

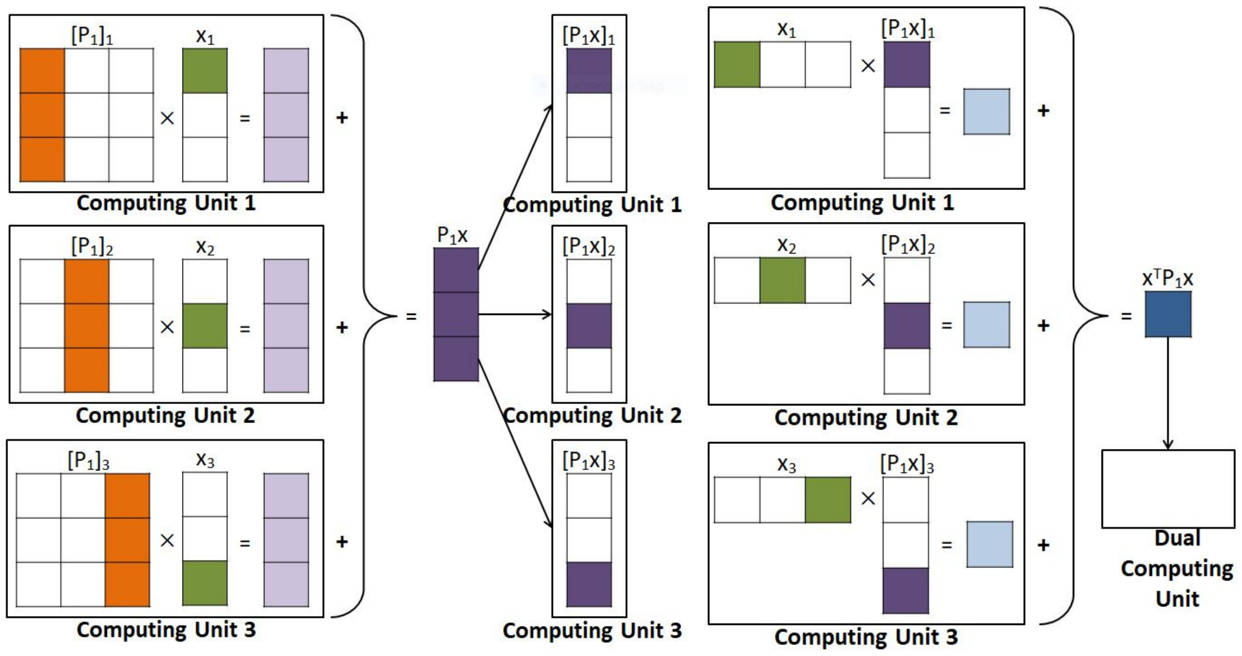
In this section, we discuss how to efficiently implement the PC^{2}PM algorithm, especially within a distributed framework.

3.4.1 Distributed Storage of Data and Parallel Computing

As mentioned in the introduction section, one key feature of the PC^{2}PM algorithm for solving convex QCQPs is that when implemented across multiple computing units, each computing unit does not need to store entire matrices. Instead, only each primal computing unit needs to store certain columns of the matrices (that is, the Hessian matrices in the objective function and the constraints). To illustrate this point, we use the primal predictor update (3.8a) as an example. Assume that ideally we have n_1 primal computing units dedicated to updating y_{j}, $\mathrm{j} = 1, \ldots, n_1$. To ease the argument, we write out the updating rule again here:

$$y_{\mathrm{j}}^{k+1} = \Pi_{\mathbb{X}_{\mathrm{j}}}\left(x_{\mathrm{j}}^k - \rho\left[P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\left(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\right) + A^T\boldsymbol{\gamma}^k\right]_{\mathrm{j}}\right). \tag{3.20}$$

In each unit j, only the values of x_{j}^k, $[P_{\mathrm{i}}]_{\mathrm{j}}$, $[\mathbf{q}_{\mathrm{i}}]_{\mathrm{j}}$ for $\mathrm{i} = 0, 1, \ldots, m_1$ and $[A]_{\mathrm{j}}$ are needed to be stored locally. To calculate $[P_{\mathrm{i}}\mathbf{x}^k]_{\mathrm{j}}$ for $\mathrm{i} = 0, 1, \ldots, m_1$, there is no need to store the entire P_{i} matrices on each computing unit. Instead, the value of $[P_{\mathrm{i}}\mathbf{x}^k]_{\mathrm{j}}$ can be obtained using MPI to communicate among all primal computing units, each of which has one column of the P_{i} matrices (and x_{j}^k) stored locally. Here we use a simple example to illustrate the mechanism. Let $n_1 = 3$, Fig 3.1a shows how $[P_1\mathbf{x}^k]_{\mathrm{j}}$, $\mathrm{j} = 1, 2, 3$ are calculated in a distributed fashion through MPI. First, each computing unit j completes a subtask of multiplying $[P_1]_{\mathrm{j}}$ and x_{j}^k using their locally stored information; then the intermediate results are summed up using the MPI_Reduce function in a root process to get the value of $P_1\mathbf{x}^k$. Each component of the vector $P_1\mathbf{x}^k$ is then sent back to the corresponding computing unit j using the MPI_Scatter function. After obtaining the values of $[P_{\mathrm{i}}\mathbf{x}^k]_{\mathrm{j}}$ for $\mathrm{i} = 0, 1, \ldots, m_1$ in this way, the update step (3.20) can be carried out upon receiving the values of $(\lambda_1^k, \ldots, \lambda_{m_1}^k)$ and $\boldsymbol{\gamma}^k$ from other dual computing units dedicated for updating the dual variables using MPI_Send and MPI_Recv



(a) Calculating $[P_1\mathbf{x}^k]_{\mathrm{j}}$ for each computing unit j.   (b) Calculating $(\mathbf{x}^k)^T P_1 \mathbf{x}^k$.

Figure 3.1. Illustrations of matrix-vector multiplications using MPI functions.

functions, with the fact that $[A^T\boldsymbol{\gamma}^k]_{\mathrm{j}} = [A]_{\mathrm{j}}^T\boldsymbol{\gamma}^k$. Such a feature will be particularly beneficial for solving large-scale QCQPs from real world applications, as in many such cases the number of variables (n_1 for $\mathbf{x}$ and n_2 for $\mathbf{u}$) can be enormous.

In the 3-dimension example shown in Fig 3.1a, once each $[P_1\mathbf{x}^k]_{\mathrm{j}}$ is received by computing unit j for $\mathrm{j} = 1, 2, 3$, a subtask of multiplying x_{j}^k and $[P_1\mathbf{x}^k]_{\mathrm{j}}$ is needed to calculate the value of $(\mathbf{x}^k)^T P_1 \mathbf{x}^k$ for dual update, such as in (3.3):

$$\mu_{\mathrm{i}}^{k+1} = \Pi_{\mathbb{R}_+}\left(\lambda_{\mathrm{i}}^k + \rho\left[\frac{1}{2}(\mathbf{x}^k)^T P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}^T\mathbf{x}^k + \mathbf{c}_{\mathrm{i}}^T\mathbf{u}^k + r_{\mathrm{i}}\right]\right). \tag{3.21}$$

Such a process is illustrated in Fig 3.1b, which shows that the locally-calculated intermediate results are summed up using the MPI_Reduce function and sent to the corresponding dual computing unit. Other matrix-vector (and vector-vector) multiplications in the update steps of Algorithm 4 can all be calculated in a similar fashion.[1]

[1]For more information, we refer the readers to our implementation codes programmed in C available online at https://github.com/BigRunTheory/A-Distributed-Algorithm-for-Large-scale-Convex-QCQPs.

Next, we examine the speedup of using multiple compute nodes for parallel distributed computing. We run the PC^{2}PM algorithm on a multi-node computer cluster, where each node has multiple cores (20 in our case), through MPI for communication among all parallel processes mapped to cores belonging to different nodes. For illustration purpose, we focus on a single-constraint convex QCQP:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^{n_1}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x} + \mathbf{q}_0^T\mathbf{x} + r_0 \\ \text{subject to} \quad & \frac{1}{2}\mathbf{x}^T P_1 \mathbf{x} + \mathbf{q}_1^T\mathbf{x} + r_1 \le 0, \qquad (\lambda_1) \end{aligned} \tag{3.22}$$

which does not contain the block of decision variable $\mathbf{u}$ or linear constraint $A\mathbf{x} + B\mathbf{u} = \mathbf{b}$. We test the PC^{2}PM algorithm for solving (3.22) with a randomly generated data set with $P_\text{i} \in \mathbb{R}^{n_1\times n_1}$, $\mathbf{q}_\text{i} \in \mathbb{R}^{n_1}$ and $r_\text{i} \in \mathbb{R}$ for $\text{i} = 0, 1$. The dimension n_1 is set at 1.7×10^4. Each matrix P_i is randomly generated as a symmetric PD matrix in the form of $P_\text{i} = Q^T DQ$, where $Q \in \mathbb{R}^{n_1\times n_1}$ is a randomly generated orthogonal matrix, and $D = diag(d_1, \ldots, d_{n_1})$ is a randomly generated diagonal matrix with all positive entries. Since $(d_1, \ldots, d_{n_1})$ are also the eigenvalues of each P_i, and the ratio of the largest eigenvalue d_max to the smallest one d_min is the condition number κ of matrix P_i, we set $\kappa(P_\text{i}) = 1.25$ and choose d_min and d_max to satisfy $d_\text{max}/d_\text{min} = \kappa(P_\text{i})$, and the remaining diagonal entries are uniformly drawn from the range $[d_\text{min}, d_\text{max}]$. The components of each vector $\mathbf{q}_\text{i}$ are uniformly generated from a random range, and each scalar r_i is generated as a random non-negative real number to guarantee the feasibility of the constraint sets.

Since the number of Lagrangian multipliers is 1, the number of dual computing units $n_\text{dual-comp}$ is also fixed as 1. The tasks of updating n_1 components of the primal decision variables $\mathbf{x}$ and $\mathbf{y}$ are evenly distributed among all the primal computing units with the number $n_\text{primal-comp}$ varying from 1 to 256 for comparison purpose. Each computing unit occupies a single core; hence the total number of cores used is equal to $n_\text{primal-comp} + n_\text{dual-comp}$. The number of nodes needed is calculated as $n_\text{node} = \lceil n_\text{core}/20 \rceil$ (where 20 is the number of cores per node). The elapsed wall-clock time T with a tolerance τ equal to 10^{-3} and 10^{-6}, respectively, is listed in Table 3.1, along with the calculated objective function values. (The specific stopping criteria are given in Section 3.4.3.)

Table 3.1. Elapsed clock time used by PC^{2}PM for solving the single-constraint convex QCQP (3.22).

PC^{2}PM using multiple nodes (max. 20 cores per node)			
$\mathbf{n_{node}}$	$\mathbf{n_{core}}$	**time** $\tau = 10^{-3}$	**time** $\tau = 10^{-6}$
1	$1+1$	5.53 h	13.84 h
1	$2+1$	2.59 h	6.46 h
1	$4+1$	1.60 h	3.86 h
1	$8+1$	1.01 h	2.50 h
1	$16+1$	0.77 h	1.86 h
2	$32+1$	0.55 h	1.37 h
4	$64+1$	0.45 h	1.08 h
7	$128+1$	0.42 h	1.02 h
14	$256+1$	0.47 h	1.01 h
obj. val.		-498.544033	-498.543200

The computational speedup S is defined as the ratio of the elapsed run time taken by a serial code to that taken by a parallel code for solving the same problem. More specifically, S is defined as:

$$S := \frac{T(1+1)}{T(n_{\text{primal-comp}}+1)}, \quad n_{\text{primal-comp}} \geq 2. \tag{3.23}$$

The speedup of solving (3.22) is shown in Fig 3.2. For this specific case, parallel computing achieved linear speedup initially. However, due to communication overhead, the speedup plateaued (or even decreased) when the number of computing units is too high. As such, we suggest that a proper number of computing units needs to be carefully chosen when implementing the PC^{2}PM algorithm to balance between computation speedup and communication overhead.

3.4.2 Adaptive Step Size with Auto-learned Allocation Weights

In establishing the global convergence of the PC^{2}PM algorithm, it is not specified how the values of ϵ_s, $s = 1, \ldots, 8$ are chosen in order to calculate the eight components $\rho_1 - \rho_8$. Here we develop a practical rule to help determine the values of ϵ_s's along the iterations. The rule may also help accelerate the algorithm's performance, based on our numerical experiments.

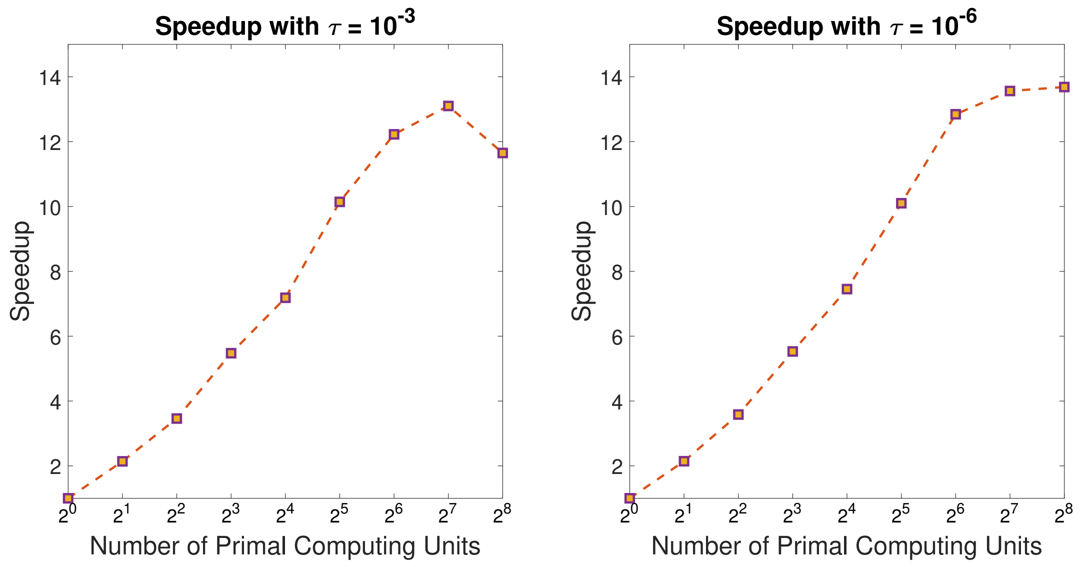


Figure 3.2. Computational speedup of PC^{2}PM for solving a single-constraint convex QCQP (3.22).

Generally speaking, for first-order algorithms, of which the PC^{2}PM algorithm also belongs to, the larger value a step size could take, the fewer number of iterations the algorithms would take to converge. For the step-size formula (3.19), it is easy to observe that the value of each ρ_s increases when the corresponding ϵ_s increases. However, the ϵ_s's cannot be too large as their summation is bounded by $1 - \epsilon_0 \leq 1$. A naive way to allocate the value of each ϵ_s is to evenly distribute the upper bound of their summation; that is, $\epsilon_s = \frac{1}{8}(1 - \epsilon_0)$ for $s = 1, \ldots, 8$, throughout the iterations. Alternatively, we introduce a weight variable $w_s > 0$ for each ϵ_s. At the beginning of the algorithm, they are all initialized to 1, indicating an even allocation of the weights. At each iteration $k = 1, 2, \ldots$, we calculate the values of $\epsilon_1^k, \ldots, \epsilon_8^k$ based on the following formulation:

$$\epsilon_s^k = \frac{w_s^k}{\sum_{s=1}^{8} w_s^k}(1 - \epsilon_0), \quad s = 1, \ldots, 8. \tag{3.24}$$

After the step size ρ^{k+1} is determined by $\min\{\rho_1^k, \ldots, \rho_8^k\}$ according to (3.19), the values of the weights $w_1^{k+1}, \ldots, w_8^{k+1}$ are updated according to the ratio of ρ^{k+1} to each ρ_s^k:

$$w_s^{k+1} = \frac{\rho^{k+1}}{\rho_s^k} w_s^k, \quad s = 1, \ldots, 8. \tag{3.25}$$

The idea of the above updating rule is to make sure that all the values of the ρ_s's will have a chance to be increased, avoiding the possibility that a particularly small ρ_s would always be chosen to determine the step size ρ^{k+1}, which would slow down the whole algorithm.

For illustration purpose, we use the PC^{2}PM algorithm with the step-size updating rule of (3.19), (3.24) and (3.25) to solve the same single-constraint convex QCQP (3.22) in the previous subsection. We test the algorithm on a data set of matrix P_{i}, vector $\mathbf{q}_{\mathrm{i}}$ and scalar r_{i} randomly generated in the same way as in the previous subsection for $\mathrm{i} = 0, 1$, but with $n_1 = 1024$, and implement it on a single core as a series code. Since it contains neither the decision variable $\mathbf{u}$ nor the linear equality constraint $A\mathbf{x} + B\mathbf{u} = \mathbf{b}$, only $\rho_1^k, \ldots, \rho_5^k$ need to be calculated at each iteration. We compare the performance of the algorithm using equally allocated weights versus using the adaptive weight updating in (3.25). The number of iterations and the elapsed wall-clock time used by the algorithm to converge with a tolerance $\tau = 10^{-3}$ are listed in Table 3.2. We also test using different values of ϵ_0, including a fixed

Table 3.2. Number of iterations and elapsed wall-clock time used by PC^{2}PM for solving a single-constraint convex QCQP (3.22) with different settings of $(\epsilon_0, \ldots, \epsilon_5)$.

	Equal Weight Allocation		**Adaptive Weight Allocation**	
Value of ϵ_0	**num. iter.**	**time**	**num. iter.**	**time**
0.5	23619	598 s	10742	272 s
10^{-1}	13119	331 s	5972	151 s
10^{-2}	11926	301 s	5430	138 s
10^{-3}	11819	300 s	5381	137 s
10^{-4}	11808	297 s	5376	136 s
10^{-5}	11807	298 s	5375	136 s
10^{-6}	11807	298 s	5375	136 s
0	11807	289 s	5375	131 s
diminishing	12024	303 s	5519	139 s

value varying from 0.5, $10^{-1}, \ldots, 10^{-6}$ to 0 and a diminishing value of $\frac{1}{\sqrt{k+1}}$. The numerical results of this specific instance suggest that by using auto-learned allocation weights, the number of iterations is cut by more than half. Comparing each rows, we also observe that the smaller the value of ϵ_0 is, the faster the algorithm converges.

3.4.3 Stopping Criteria

Since we consider a convex QCQP and assume that the Slater's CQ holds, the first-order optimality conditions (aka the KKT conditions) are both necessary and sufficient. More specifically, for an optimal solution $(\mathbf{x}^*, \mathbf{u}^*)$ of the QCQP (3.2) and its corresponding dual solution $(\boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$, the following conditions are satisfied:

$$-\Big[P_0\mathbf{x}^* + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^*(P_{\mathrm{i}}\mathbf{x}^* + \mathbf{q}_{\mathrm{i}}) + A^T\boldsymbol{\gamma}^*\Big]_{\mathrm{j}} \in \mathcal{N}_{\mathbb{X}_{\mathrm{j}}}(x_{\mathrm{j}}^*), \quad \mathrm{j} = 1, \dots, n_1 \tag{3.26}$$

$$\mathbf{c}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^*\mathbf{c}_{\mathrm{i}} + B^T\boldsymbol{\gamma}^* = \mathbf{0} \tag{3.27}$$

$$0 \le \lambda_{\mathrm{i}}^* \perp -\frac{1}{2}(\mathbf{x}^*)^T P_{\mathrm{i}}\mathbf{x}^* - \mathbf{q}_{\mathrm{i}}^T\mathbf{x}^* - \mathbf{c}_{\mathrm{i}}^T\mathbf{u}^* - r_{\mathrm{i}} \ge 0, \quad \mathrm{i} = 1, \dots, m_1 \tag{3.28}$$

$$A\mathbf{x}^* + B\mathbf{u}^* - \mathbf{b} = \mathbf{0}. \tag{3.29}$$

Conversely, any primal-dual pair $(\mathbf{x}^*, \mathbf{u}^*; \boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$ satisfying the above equations is optimal to the primal and dual of the QCQP (3.2), respectively.

Based on the optimality conditions (3.26) – (3.29), we choose stopping criteria for our algorithm to measure both the primal and dual feasibility, as well as complementarity. More specifically, at each iteration k, we measure the following two residuals:

$$res_1^k = \sqrt{\frac{1}{n_1 + n_2}\Big[\sum_{\mathrm{j}=1}^{n_1}\Big(res_1^k\_x_{\mathrm{j}}\Big)^2 + \|\mathbf{c}_0 + \sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\mathbf{c}_{\mathrm{i}} + B^T\boldsymbol{\gamma}^k\|_2^2\Big]}, \text{ and} \tag{3.30}$$

$$res_2^k = \sqrt{\begin{array}{l}\displaystyle\frac{1}{m_1 + m_2}\Big[\sum_{\mathrm{i}=1}^{m_1}\Big[\lambda_{\mathrm{i}}^k\Big|\frac{1}{2}(\mathbf{x}^k)^T P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}^T\mathbf{x}^k + \mathbf{c}_{\mathrm{i}}^T\mathbf{u}^k + r_{\mathrm{i}}\Big|\Big]^2 \\ + \|A\mathbf{x}^k + B\mathbf{u}^k - \mathbf{b}\|_2^2\Big]\end{array}}, \tag{3.31}$$

where $res_1^k\_x_{\mathrm{j}}$ in (3.30) depends on the actual form of the constraint set $\mathbb{X}$. Again, for box constrains $0 \le x_{\mathrm{j}} \le \bar{X}_{\mathrm{j}}$, $\mathrm{j} = 1, \dots, n_1$, we have that

$$res_1^k\_x_{\mathrm{j}} : = \begin{cases} \min\{0, \big[\mathbf{grad}_{\mathbf{x}}\big]_{\mathrm{j}}\}, & \text{if } x_{\mathrm{j}}^k = 0 \\ \big[\mathbf{grad}_{\mathbf{x}}\big]_{\mathrm{j}}, & \text{if } 0 < x_{\mathrm{j}}^k < \bar{X}_{\mathrm{j}} \\ \max\{0, \big[\mathbf{grad}_{\mathbf{x}}\big]_{\mathrm{j}}\}, & \text{if } x_{\mathrm{j}}^k = \bar{X}_{\mathrm{j}}, \end{cases} \tag{3.32}$$

where $\mathbf{grad}_{\mathbf{x}} = P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1} \lambda_{\mathrm{i}}^k(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}) + A^T\boldsymbol{\gamma}^k$. We terminate our algorithm when both of the two residuals drop below a pre-specified tolerance. Note that the residuals defined in (3.30) and (3.31) are based on the average residuals of all the constraints. Other forms of residual metric, such as using the maximum residuals of all the constraints, can also be used.

3.5 Numerical Experiments

In this section, we present more numerical results for solving large-scale convex QCQPs using our algorithm. We first conduct numerical experiments of applying the PC^{2}PM algorithm to solve convex QCQPs of the standard form (3.1), with randomly generated data sets of various sizes. We then solve convex QCQPs with explicit linear constraints as in (3.2), which naturally arise from multiple kernel learning applications. For both sets of experiments, we compare the performance of our algorithm with the current state-of-the-art commercial solver CPLEX 12.8.0, which uses the barrier optimizer for solving convex QCQPs. We implement PC^{2}PM with multiple compute nodes on Purdue University's Rice cluster through MPI, called from a C program. Each compute node on the cluster has two 10-core Intel Xeon-E5 processors (that is, 20 cores per node) and 64 GB of memory. CPLEX 12.8.0 is also called using a C program and implemented on a single compute node (with 20 cores). Note that CPLEX alone, as a centralized algorithm, cannot be run on multiple compute nodes through MPI, but it does allow multiple parallel threads that can be invoked by the barrier optimizer. More specifically, CPLEX has a parameter, CPXPARAM_Threads, to call for multithread computing [35]. When CPXPARAM_Threads is set to be 1, CPLEX is single threaded; when it is set to be 0, CPLEX can use up to 32 threads, or the number of cores of the machine (with each core being a thread), whichever is smaller. In our experiments, we always set CPXPARAM_Threads as 0, which gives CPLEX 20 threads (since each of our compute node has 20 cores).

3.5.1 Solving Standard-Form Convex QCQPs

We first apply PC^{2}PM to solve convex QCQPs of the standard form (3.1), without the decision variables $\mathbf{u}$ or the explicit linear constraints $A\mathbf{x} + B\mathbf{u} - \mathbf{b} = \mathbf{0}$. The input data consist of matrix P_i, vector $\mathbf{q}_\mathrm{i}$ and scalar r_i for $\mathrm{i} = 0, 1, \ldots, m_1$, all of which are randomly generated in the same way as in Section 3.4.1. The only difference here is that instead of keeping the matrices' condition number fixed, we increase it from 1.25 to 100 to all matrices. The decision variable's dimension n_1 is set at 1.7×10^4, and the number of constraints m_1 increases from 1 to 16.

To balance between the computation speedup and communication overhead, we implement our algorithm with 128 cores allocated for primal variables' updating: (3.8a) (3.10a), (3.8b) and (3.10b), and m_1 (the number of quadratic constraints) cores for dual updating: (3.3) and (3.11). The total number of compute nodes needed is calculated as $n_\text{node} = \lceil \frac{n_\text{core}}{20} \rceil$ $= \lceil \frac{128+m}{20} \rceil$. The stopping criteria we use are defined in (3.30) and (3.31), with the tolerance set at $\tau = 10^{-3}$. Table 3.3 reports the elapsed wall-clock time used by the PC^{2}PM algorithm, along with the maximal amount of memory used by each compute node and the final objective function value, with respect to increasing condition number κ. The performance of CPLEX 12.8.0 with the same convergence tolerance is also presented in Table 3.3 for comparison. In the first group of tests with $m_1 = 1$, our algorithm converges much faster than CPLEX, and also uses much less memory. For the rest groups of test cases, CPLEX fails to provide a solution (actually fails to complete even a single iteration) due to running out of memory; while PC^{2}PM still converges within a reasonable amount of time. As the scale of the problem increases, our algorithm exhibits favorable scalability, due to its distributed storage of data and the capability of massively parallel computing. Another interesting observation from Table 3.3, though we do not know the underlying reason, is that the when the number of quadratic constraints (m_1) is small, PC^{2}PM's run time appears to be sensitive to the condition number of matrices (i.e., the Hessian matrices of the objective function and the constraints); yet when m_1 becomes larger, the effect of condition numbers on the run time appears to be subdued.

We also plot the two residuals res_1^k and res_2^k in Fig 3.3, with res_1^k corresponding to the

Table 3.3. Comparison of PC^{2}PM with CPLEX 12.8.0 for solving standard-form, large-scale convex QCQPs.

$\mathbf{n_1}$	$\mathbf{m_1}$	$\boldsymbol{\kappa}$		$\mathbf{n_{node}}$	$\mathbf{n_{core}}$	**max. mem. /node**	**time**	**obj. val.**
1.7×10^4	1	1.25	PC^{2}PM ($\tau^{\text{PC}^2\text{PM}} = 10^{-3}$)	7	$128+1$	2.1 GB	0.37 h	-418.621946
		10					0.69 h	-425.943254
		100					3.87 h	-420.620978
		1.25	CPLEX 12.8.0 ($\tau^{\text{Barrier}} = 10^{-3}$)	1	20	40.6 GB	6.42 h	-418.559064
		10					6.15 h	-425.947242
		100					6.17 h	-420.644988
1.7×10^4	2	1.25	PC^{2}PM ($\tau^{\text{PC}^2\text{PM}} = 10^{-3}$)	7	$128+2$	2.5 GB	0.77 h	-326.237771
		10					0.85 h	-330.950122
		100					2.10 h	-322.231765
		1.25	CPLEX 12.8.0 ($\tau^{\text{Barrier}} = 10^{-3}$)	1	20	N.A.	N.A.	N.A.
		10						
		100						
1.7×10^4	4	1.25	PC^{2}PM ($\tau^{\text{PC}^2\text{PM}} = 10^{-3}$)	7	$128+4$	3.1 GB	1.03 h	-250.572460
		10					1.13 h	-254.314644
		100					1.28 h	-243.154365
		1.25	CPLEX 12.8.0 ($\tau^{\text{Barrier}} = 10^{-3}$)	1	20	N.A.	N.A.	N.A.
		10						
		100						
1.7×10^4	8	1.25	PC^{2}PM ($\tau^{\text{PC}^2\text{PM}} = 10^{-3}$)	7	$128+8$	4.5 GB	1.83 h	-188.497033
		10					1.88 h	-190.964860
		100					2.20 h	-189.945073
		1.25	CPLEX 12.8.0 ($\tau^{\text{Barrier}} = 10^{-3}$)	1	20	N.A.	N.A.	N.A.
		10						
		100						
1.7×10^4	16	1.25	PC^{2}PM ($\tau^{\text{PC}^2\text{PM}} = 10^{-3}$)	8	$128+16$	6.5 GB	2.54 h	-148.730182
		10					2.95 h	-144.870372
		100					3.15 h	-147.309600
		1.25	CPLEX 12.8.0 ($\tau^{\text{Barrier}} = 10^{-3}$)	1	20	N.A.	N.A.	N.A.
		10						
		100						

gradient of the Lagrangian function, and res_2^k corresponding to the feasibility and complementarity conditions. The three plots in a same row are with the same number of constraints m_1, but different condition numbers of the Hessian matrices. As seen in Fig 3.3, from left to right, as the condition number κ increases, more iterations are required for the PC^{2}PM algorithm to converge. Another observation is that when the number of constraints increases (i.e., from top to bottom), the convergence of the residuals becomes more smooth.

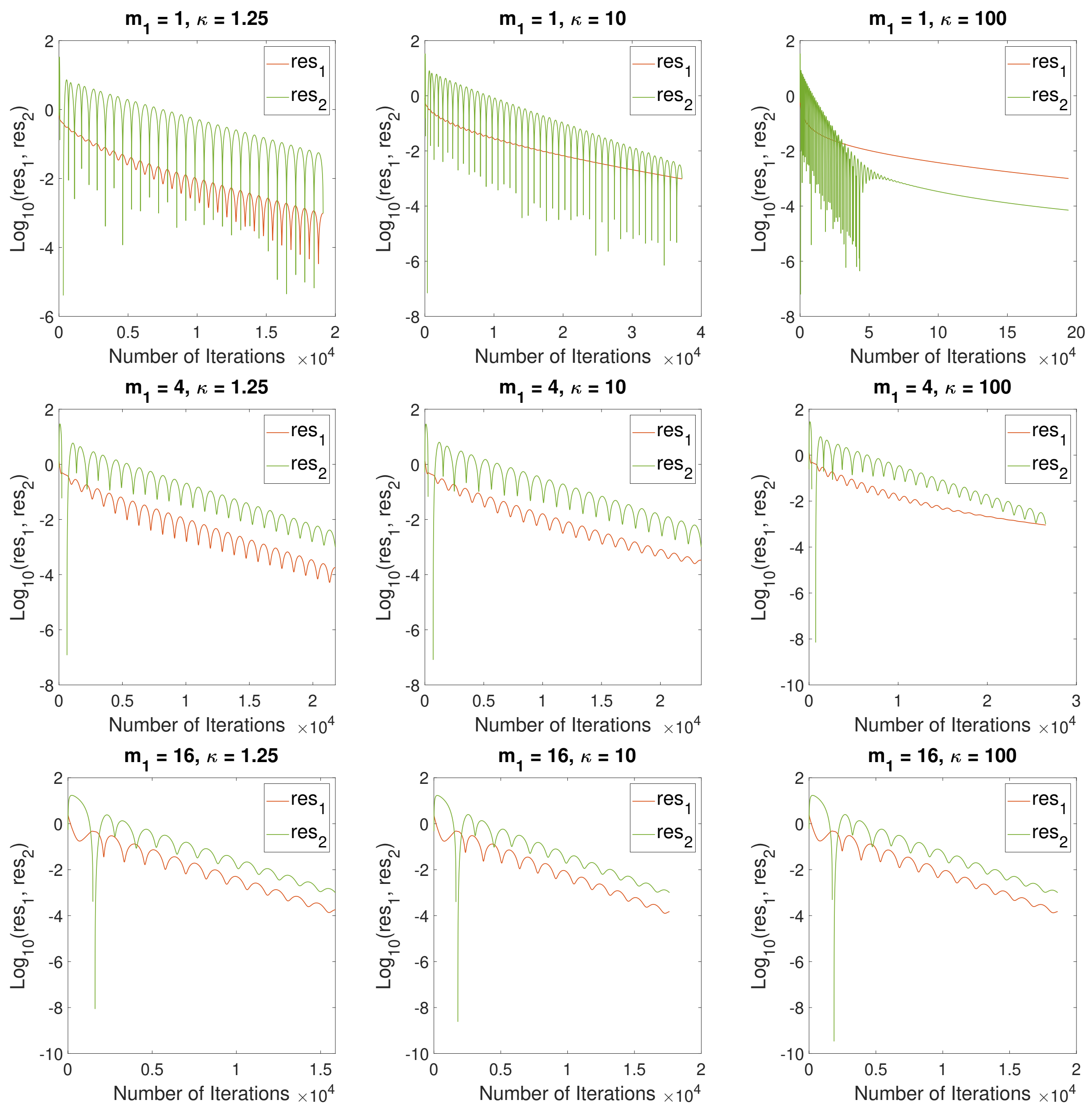


Figure 3.3. Convergence of residuals.

3.5.2 Multiple Kernel Learning in Support Vector Machine

In this subsection, we briefly introduce how the Support Vector Machine (SVM) with multiple kernel learning can be formulated as a convex QCQP, and present numerical results of using our algorithm to solve large-scale instances. As discussed in [36], SVM is a discriminative classifier proposed for binary classification problems. Given a set of n_{tr} pairs of independently and identically distributed training data points $\{(\mathbf{d}_{\mathrm{j}}, l_{\mathrm{j}})\}_{\mathrm{j}=1}^{n_{tr}}$, where $\mathbf{d}_{\mathrm{j}} \in \mathbb{R}^{n_d}$ is the n_d-dimensional input vector and $l_{\mathrm{j}} \in \{-1, 1\}$ is its class label, SVM searches for a hyperplane that can best separate the points from two classes. The hyperplane is defined as $\{\mathbf{d} \in \mathbb{R}^{n_d} | f(\mathbf{d}) = \boldsymbol{\beta}^T \mathbf{d} + \beta_0 = 0\}$, where $\boldsymbol{\beta} \in \mathbb{R}^{n_d}$ is a unit vector with $\|\boldsymbol{\beta}\|_2 = 1$, and $\beta_0 \in \mathbb{R}$ is a scalar. The points belonging to either class should be separated as far away from the hyperplane as possible, while still remain on the correct side. When the data points cannot be clearly separated in the original space $\mathbb{R}^{n_d}$, we instead search in a feature space $\mathbb{R}^{n_f}$, and map the input data space $\mathbb{R}^{n_d}$ to the feature space through a function $\Phi : \mathbb{R}^{n_d} \to \mathbb{R}^{n_f}$. For example, a 2-dimension data space can be lifted to a 3-dimension feature space. Using the function mapping Φ, we can define a *kernel function* $k : \mathbb{R}^{n_d} \times \mathbb{R}^{n_d} \to \mathbb{R}$ as $k(\mathbf{d}, \mathbf{d}') :=< \Phi(\mathbf{d}),\ \Phi(\mathbf{d}') >$ for any $\mathbf{d}, \mathbf{d}' \in \mathbb{R}^{n_d}$, where $<, >$ denotes an inner product. The resulting discriminant function $\mathcal{G} : \mathbb{R}^{n_d} \to \{-1, 1\}$ that SVM searches for can be expressed as:

$$\mathcal{G}(\mathbf{d}) = \mathrm{sign}\Big(\sum_{\mathrm{j}=1}^{n_{tr}} \alpha_{\mathrm{j}} l_{\mathrm{j}} k(\mathbf{d}_{\mathrm{j}}, \mathbf{d}) + b \Big), \quad \forall \mathbf{d} \in \mathbb{R}^{n_d}, \tag{3.33}$$

where $\boldsymbol{\alpha} \equiv (\alpha_1, \ldots, \alpha_{n_{tr}})^T$ is the weight vector and b is the bias. The popular choices of kernel functions in the SVM literature include the linear kernel function k_{LIN}, the polynomial kernel function k_{POL} and the Gaussian kernel function k_{GAU}:

$$k_{LIN}(\mathbf{d}, \mathbf{d}') := \mathbf{d}^T \mathbf{d}', \quad \forall \mathbf{d}, \mathbf{d}' \in \mathbb{R}^{n_d} \tag{3.34a}$$

$$k_{POL}(\mathbf{d}, \mathbf{d}') := (1 + \mathbf{d}^T \mathbf{d}')^2, \quad \forall \mathbf{d}, \mathbf{d}' \in \mathbb{R}^{n_d} \tag{3.34b}$$

$$k_{GAU}(\mathbf{d}, \mathbf{d}')) := \mathrm{e}^{-\frac{\|\mathbf{d}-\mathbf{d}'\|_2^2}{2\sigma^2}}, \quad \sigma > 0, \forall \mathbf{d}, \mathbf{d}' \in \mathbb{R}^{n_d}. \tag{3.34c}$$

Instead of using a single kernel function, [21] explores SVM using a kernel function that is expressed as a non-negative combination of a pre-specified set of kernel functions $\{k_1, \ldots, k_m\}$, with the non-negative coefficients $\lambda_1, \ldots, \lambda_m$ to be allocated; that is, $k(\mathbf{d}, \mathbf{d}') = \sum_{\mathrm{i}=1}^{m} \lambda_{\mathrm{i}} k_{\mathrm{i}}(\mathbf{d}, \mathbf{d}')$ for any $\mathbf{d}, \mathbf{d}' \in \mathbb{R}^{n_d}$ with $\lambda_1, \ldots, \lambda_m \geq 0$. The allocation process can be expressed as solving a convex QCQP, where each λ_{i} is the Lagrangian multiplier corresponding to each quadratic constraint. The formulation of the convex QCQP, as provided in [21], is as follows:

(i) **1-norm Soft Margin SVM** learns the coefficients through solving the following convex QCQP:

$$\begin{aligned} \underset{\boldsymbol{\alpha}\in\mathbb{R}^{n_{tr}},\alpha_0\in\mathbb{R}}{\text{minimize}} \quad & -\mathbf{e}^T\boldsymbol{\alpha} + R\alpha_0 \\ \text{subject to} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\Big[\frac{1}{R_{\mathrm{i}}}G_{\mathrm{i}}(K_{\mathrm{i},tr})\Big]\boldsymbol{\alpha} - \alpha_0 \le 0, \quad \mathrm{i}=1,\ldots,m, \qquad (\lambda_{\mathrm{i}}) \\ & \sum_{\mathrm{j}=1}^{n_{tr}} l_{\mathrm{j}}\alpha_{\mathrm{j}} = 0, \qquad (\gamma) \\ & 0 \le \alpha_{\mathrm{j}} \le C, \quad \mathrm{j}=1,\ldots,n_{tr}, \end{aligned} \tag{3.35}$$

(ii) **2-norm Soft Margin SVM** learns the coefficients through solving the following convex QCQP:

$$\begin{aligned} \underset{\boldsymbol{\alpha}\in\mathbb{R}_+^{n_{tr}},\alpha_0\in\mathbb{R}}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\Big[\frac{1}{C}I_{n_{tr}}\Big]\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} + R\alpha_0 \\ \text{subject to} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\Big[\frac{1}{R_{\mathrm{i}}}G_{\mathrm{i}}(K_{\mathrm{i},tr})\Big]\boldsymbol{\alpha} - \alpha_0 \le 0, \quad \mathrm{i}=1,\ldots,m, \qquad (\lambda_{\mathrm{i}}) \\ & \sum_{\mathrm{j}=1}^{n_{tr}} l_{\mathrm{j}}\alpha_{\mathrm{j}} = 0, \qquad (\gamma) \end{aligned} \tag{3.36}$$

where the vector $\mathbf{e}$ denotes an n_{tr}-dimensional vector of all ones. Given a labeled training data set $\mathcal{S}_{tr} = \{(\mathbf{d}_{\mathrm{j}}, l_{\mathrm{j}})\}_{\mathrm{j}=1}^{n_{tr}}$ and an unlabeled test data set $\mathcal{S}_t = \{\mathbf{d}_{\mathrm{j}}\}_{\mathrm{j}=1}^{n_t}$, a matrix $K_{\mathrm{i}} \in \mathbb{R}^{(n_{tr}+n_t)\times(n_{tr}+n_t)}$ can be defined on the entire data set $\mathcal{S}_{tr} \cup \mathcal{S}_t$ as

$$K_{\mathrm{i}} := \begin{pmatrix} K_{\mathrm{i},tr} & K_{\mathrm{i},(tr,t)} \\ K_{\mathrm{i},(tr,t)}^T & K_{\mathrm{i},t} \end{pmatrix}. \tag{3.37}$$

The submatrix $K_{\mathrm{i},tr} \in \mathbb{R}^{n_{tr}\times n_{tr}}$ is a square symmetric matrix, whose jj′-th element is directly defined by a kernel function: $[K_{\mathrm{i},tr}]_{\mathrm{jj}'} := k_{\mathrm{i}}(\mathbf{d}_{\mathrm{j}}, \mathbf{d}_{\mathrm{j}'})$ for any $\mathbf{d}_{\mathrm{j}}, \mathbf{d}_{\mathrm{j}'}$ in $\mathcal{S}_{tr}$. The submatrices $K_{\mathrm{i},(tr,t)} \in \mathbb{R}^{n_{tr}\times n_t}$ and $K_{\mathrm{i},t} \in \mathbb{R}^{n_t\times n_t}$ are defined in the same way but with different input vectors. The matrix $G_{\mathrm{i}}(K_{\mathrm{i},tr}) \in \mathbb{R}^{n_{tr}\times n_{tr}}$ in the quadratic constraint of (3.35) and (3.36) is a square symmetric matrix with its jj′-th element being $[G_{\mathrm{i}}(K_{\mathrm{i},tr})]_{\mathrm{jj}'} = l_{\mathrm{j}} l_{\mathrm{j}'} [K_{\mathrm{i},tr}]_{\mathrm{jj}'}$. Note that each kernel matrix $K_{\mathrm{i},tr}$ is a symmetric PSD matrix (see Proposition 2 in [21]), then each $G_{\rm i}(K_{{\rm i},tr})$ is also a symmetric PSD matrix, since $G_{\rm i}(K_{{\rm i},tr}) = LK_{{\rm i},tr}L$, where $L \coloneqq diag(l_1, \ldots, l_{n_{tr}})$. Let $R_{\rm i}$ denote trace$(K_{\rm i})$ for i $= 1, \ldots, m$, and $R = \sum_{\rm i=1}^{m} \lambda_{\rm i} R_{\rm i}$ can be fixed as a given number. The parameter C is a fixed positive scalar from the soft margin criteria.

Once the optimal primal-dual solution $(\boldsymbol{\alpha}^*; \lambda_1^*, \ldots, \lambda_m^*)$ is found from either (3.35) or (3.36), combining with those pre-specified $k_{\rm i}$'s, it can be used to label the test data set according to the following discriminant function $\mathcal{G}_{\rm MKL} : \mathbb{R}^{n_d} \to \{-1, 1\}$:

$$\mathcal{G}_{\rm MKL}(\mathbf{d}_{\rm j'}) = \text{sign}\left(\sum_{\rm j=1}^{n_{tr}} \alpha_{\rm j}^* l_{\rm j} \Big[\sum_{\rm i=1}^{m} \lambda_{\rm i}^* k_{\rm i}(\mathbf{d}_{\rm j}, \mathbf{d}_{\rm j'}) \Big] + b \right), \quad \forall \mathbf{d}_{\rm j'} \in \mathcal{S}_t. \tag{3.38}$$

Compared with (3.33), the only difference is the replacement of a non-negative combination of $k_{\rm i}$'s with coefficients $\lambda_1^*, \ldots, \lambda_m^*$. The test set accuracy (TSA) can then be obtained by measuring the percentage of the test data points accurately labeled according to the function (3.38).

The formulation (3.35) and (3.36) provide instances of convex QCQPs in the form of (3.2), and we apply the PC^{2}PM to solve them. The input data set is artificially generated as specified in [37], which is also used in [21]; however, our data set has a much larger size than in [21]. We first generate $6,880$ data points, with each data point a 20-dimension vector, which are drawn from a multivariate normal distribution with a unit covariance matrix and the mean of $(a, \ldots, a)$. These data points form the first class that are all labeled with 1. Another $6,880$ points of 20-dimension vectors are drawn from another multivariate normal distribution with a unit covariance matrix and the mean of $(-a, \ldots, -a)$, which form the second class that are all labeled with -1. The value of a is set as 3.0. We use a set of pre-specified kernel functions $\{k_1, \ldots, k_5\}$ that contains all Gaussian kernel functions defined in (3.34c) whose σ^2 equal to 0.01, 0.1, 1, 10 and 100 respectively. Each matrix $K_{\rm i}$ is normalized and $R_{\rm i}$ = trace$(K_{\rm i})$ is set to be 1.0 for i $= 1, \ldots, 5$. Then $R = \sum_{\rm i=1}^{5} \lambda_{\rm i} R_{\rm i} = \sum_{\rm i=1}^{5} \lambda_{\rm i}$, is restricted to be 5.0. The value of C is set at 2.0 for both two soft margin SVMs. Numerical results of both 1-norm and 2-norm soft margin SVMs using the above five kernel functions are summarized in Table 3.4. The data set of a total number of $6,880 + 6,880 = 13,760$

Table 3.4. Comparison of PC^{2}PM with CPLEX 12.8.0 for solving multiple kernel learning problems using 5 Gaussian kernel functions.

Rand. Part.	SVM		max. mem. /node	time	λ_1^*	λ_2^*	λ_3^*	λ_4^*	λ_5^*	TSA
1	SM1	PC^{2}PM	2.0 GB	4.84 h	0.00	0.00	0.00	0.00	5.05	99.75%
	SM2			3.31 h	0.00	0.00	0.00	0.00	5.03	99.85%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2		52.0 GB	2.12 h	0.00	0.00	0.00	0.35	4.64	99.85%
2	SM1	PC^{2}PM	2.0 GB	5.44 h	0.00	0.00	0.00	0.00	5.07	99.89%
	SM2			2.85 h	0.00	0.00	0.00	0.00	5.03	99.93%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2		52.0 GB	2.08 h	0.00	0.00	0.00	0.35	4.65	99.93%
3	SM1	PC^{2}PM	2.0 GB	5.88 h	0.00	0.00	0.00	0.00	5.01	99.67%
	SM2			3.04 h	0.00	0.00	0.00	0.00	5.03	99.89%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2		52.0 GB	2.15 h	0.00	0.00	0.00	0.37	4.62	99.89%
4	SM1	PC^{2}PM	2.0 GB	5.22 h	0.00	0.00	0.00	0.00	5.07	99.60%
	SM2			2.92 h	0.00	0.00	0.00	0.00	5.03	99.89%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2		52.0 GB	2.30 h	0.00	0.00	0.00	0.35	4.64	99.89%
5	SM1	PC^{2}PM	2.0 GB	5.52 h	0.00	0.00	0.00	0.00	5.03	99.78%
	SM2			2.80 h	0.00	0.00	0.00	0.00	5.03	99.85%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2		52.0 GB	2.21 h	0.00	0.00	0.00	0.34	4.66	99.85%

data points is randomly partitioned into 80% for training and 20% for testing. The reported results are over five different random partitions.

We implement PC^{2}PM using 128 cores for primal updates and 5 cores for dual updates, which amount to a total of 7 compute nodes on Purdue's Rice cluster. The elapsed wall-clock time used by PC^{2}PM to converge with a tolerance $\tau = 10^{-3}$ is presented in Table 3.4, along with the maximal amount of memory used by each node. We also report in Table 3.4 the learned non-negative coefficients $\lambda_1^*, \dots \lambda_5^*$, as well as the TSA. The performance of CPLEX 12.8.0 with the same tolerance is also presented in Table 3.4 for comparison. As shown by the value of the coefficients learned, the Gaussian kernel function k_5 with $\sigma^2 = 100.0$ is selected by the models of both two soft margin SVMs. For 2-norm soft margin SVM, while PC^{2}PM uses more time to converge than CPLEX, it uses much less memory (as expected). For TSA, both PC^{2}PM and CPLEX obtain the same value, calculated using their own optimal solution point $(\boldsymbol{\alpha}^*, \lambda_1^*, \ldots, \lambda_m^*)$. For 1-norm soft margin SVM, CPLEX fails to provide a solution due to running out of memory, while PC^{2}PM still solves the problem.

While the numerical experiments so far have demonstrated the scalability of the PC^{2}PM algorithm due to its distributed data storage and natural decomposition to facilitate parallel computing, in the following experiments, we show the benefits of the PC^{2}PM algorithm for not requiring any matrix decompositions. In this test, we use three kernel functions, instead of five, to solve (3.35) and (3.36). The three kernel functions consist of k_1 – the Gaussian kernel function with $\sigma^2 = 100.0$, k_2 – a linear kernel function defined in (3.34a), and k_3 – a polynomial kernel function defined in (3.34b). All the other settings remain the same as in the previous experiment (except for the value of R, which is set as 3.0). The numerical results are reported in Table 3.5. For all groups of tests, CPLEX returns an error stating that the

Table 3.5. Comparison of PC^{2}PM with CPLEX 12.8.0 for solving multiple kernel learning problems using 3 kernel functions. (For 1-norm soft margin SVM, PC^{2}PM converges with a tolerance $\tau = 4 \times 10^{-3}$ instead of 10^{-3}.)

Rand. Part.	SVM		max. mem. /node	time	λ_1^*	λ_2^*	λ_3^*	TSA
1	SM1	PC^{2}PM	1.7 GB	6.24 h	0.00	3.41	0.00	99.85%
	SM2			3.52 h	0.00	3.10	0.00	99.85%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2							
2	SM1	PC^{2}PM	1.7 GB	6.56 h	0.00	3.41	0.00	99.93%
	SM2			3.53 h	0.00	3.10	0.00	99.93%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2							
3	SM1	PC^{2}PM	1.7 GB	6.37 h	0.00	3.41	0.00	99.89%
	SM2			3.84 h	0.00	3.10	0.00	99.89%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2							
4	SM1	PC^{2}PM	1.6 GB	6.68 h	0.00	3.41	0.00	99.85%
	SM2			3.70 h	0.00	3.10	0.00	99.85%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2							
5	SM1	PC^{2}PM	1.7 GB	6.27 h	0.00	3.41	0.00	99.85%
	SM2			3.73 h	0.00	3.10	0.00	99.85%
	SM1	CPLEX 12.8.0	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	SM2							

quadratic constraint containing $G_3(K_{3,tr})$ is not convex, which is theoretically impossible because each matrix $G_{\mathrm{i}}(K_{\mathrm{i},tr})$ is at least a PSD matrix as we discussed previously; while PC^{2}PM solves all the instances without any issues. The error returned by CPLEX is created likely by the failure of matrix decomposition of a large-scale PSD matrix due to precision limit. Once we reduce the size of the matrices in (3.35) and (3.36), CPLEX can then solve the instances without error messages. This numerical experiment illustrates that not requiring matrix decomposition in the PC^{2}PM is not just of computational convenience; it can indeed make the algorithm more robust to solve large-scale problems without facing potential issues caused by floating point arithmetic.

3.6 Conclusion and Future Works

In this chapter, we propose a novel distributed algorithm, built upon the original idea of the PCPM algorithm, that can solve non-separable convex QCQPs in a Jacobi-fashion (that is, parallel updating). Numerical results show that our algorithm, termed as PC^{2}PM, exhibits much better scalability when compared to CPLEX, which uses the IPM to solve convex QCQPs. The superiority of algorithm's scalability is attributed to the three key features of the algorithm design: first, the PC^{2}PM algorithm can decompose primal (and dual) variables down to the scalar level and update them in parallel, even when the quadratic constraints are non-separable; second, when implementing the algorithm, only the related columns of all the Hessian matrices need to be stored locally, instead of the entire matrices on each of computing unit in a parallel computing setting; third, our algorithm does not need any matrix decomposition (unlike any semi-definite-programming-based approach), which can improve the algorithm's robustness, especially when solving convex QCQPs with PSD matrices, as demonstrated in our numerical experiments summarized in Table 3.5. The second and the third feature together make our algorithm particularly suitable to solve extreme-scale QCQPs, which likely will cause memory issues for other algorithms.

In addition to the scalability of the PC^{2}PM algorithm, its ability to solve non-separable, quadratically constrained problems in Jacobi-fashion should also be emphasized, as in general it is very difficult to design distributed algorithms with Jacobi-style update (as opposed to the sequential Gauss-Seidel update) to solve optimization problems with non-separable constraints. Whether the algorithm idea from PC^{2}PM can be extended to solve more general convex problems is certainly worth of exploring. There are several other lines of research that can be done to improve the current work. First, while we proved convergence of PC^{2}PM, we cannot prove its convergence rate as of now. The numerical results in Figure 3.3 suggest that the convergence rate is likely to be linear, but whether this is true for QCQPs with PSD Hessian matrix in the objective function is unknown. Second, while the parallel updating of the primal variables is a nice property of PC^{2}PM, it is still a synchronous algorithm in the sense that the algorithm needs to wait for all primal and dual updates to be done before it can move to the next iteration. An asynchronous implementation of the algorithm will no doubt make it even more suitable for distributed computing, and we defer it to our future work. Third, there have been increasing works on solving large-scale non-convex QCQPs. As mentioned in the introduction section, one algorithm idea is to solve it with a sequence of convexified QCQPs, where our algorithm is then applicable. This naturally leads to an algorithm with nested loops, where the outer loop lays out sequential convexification, and the inner loop invokes our algorithm. It would be interesting to see how such a nested algorithm performs in practice, especially with large-scale problems.

3.7 Proofs in Section 3.3

3.7.1 Proof of Proposition 3.3.3

We first prove the inequality (3.17). Consider the linear approximation of the Lagrangian function of a QCQP, as defined in (3.14), with a given point $\zeta^k \equiv (\mathbf{x}^k, \lambda^k, \gamma^k)$. Let $\widehat{\mathbf{z}} = (\mathbf{y}^{k+1}, \mathbf{v}^{k+1})$, the $(k+1)$-th iteration of the primal predictor of $\mathbf{x}^k$ and $\mathbf{u}^k$ in the PC^{2}PM algorithm, as given in (3.8a) and (3.10a), respectively. By Lemma 3.3.1, we know that $\widehat{\mathbf{z}}$ is the unique minimizer of the corresponding proximal minimization problem in (3.15b). By defining $\bar{\mathbf{z}} = (\mathbf{x}^k, \mathbf{u}^k)$ and $\mathbf{z} = (\mathbf{x}^{k+1}, \mathbf{u}^{k+1})$, and using Lemma 2.2.1, we have that

$$2\rho^{k+1}\Bigg[\mathcal{R}(\widehat{\mathbf{z}}; \zeta^k) - \mathcal{R}(\mathbf{z}; \zeta^k)\Bigg] \leq \|\bar{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\widehat{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\widehat{\mathbf{z}} - \bar{\mathbf{z}}\|_2^2,$$

which leads to the following expanded inequality

$$\begin{aligned}
&2\rho^{k+1}\Bigg\{(P_0\mathbf{x}^k+\mathbf{q}_0)^T\mathbf{y}^{k+1}+\mathbf{c}_0^T\mathbf{v}^{k+1}+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\Big[(P_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}})^T\mathbf{y}^{k+1}+\mathbf{c}_{\mathrm{i}}^T\mathbf{v}^{k+1}+r_{\mathrm{i}}\Big]\\
&\qquad+(\boldsymbol{\gamma}^k)^T(A\mathbf{y}^{k+1}+B\mathbf{v}^{k+1}-\mathbf{b})\Bigg\}\\
&-2\rho^{k+1}\Bigg\{(P_0\mathbf{x}^k+\mathbf{q}_0)^T\mathbf{x}^{k+1}+\mathbf{c}_0^T\mathbf{u}^{k+1}+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\Big[(P_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}})^T\mathbf{x}^{k+1}+\mathbf{c}_{\mathrm{i}}^T\mathbf{u}^{k+1}+r_{\mathrm{i}}\Big]\\
&\qquad+(\boldsymbol{\gamma}^k)^T(A\mathbf{x}^{k+1}+B\mathbf{u}^{k+1}-\mathbf{b})\Bigg\}\\
&\leq\Big(\|\mathbf{x}^k-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_2^2\Big)\\
&\quad-\Big(\|\mathbf{y}^{k+1}+\mathbf{x}^{k+1}\|_2^2+\|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2\Big)-\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^k\|_2^2+\|\mathbf{v}^{k+1}-\mathbf{u}^k\|_2^2\Big).
\end{aligned}\tag{3.39}$$

Now consider the $\mathcal{R}$ function at a different given point $\zeta^{k+1}\equiv(\mathbf{y}^{k+1},\mu^{k+1},\boldsymbol{\nu}^{k+1})$. With a slight abuse of notation, we now let $\widehat{\mathbf{z}}=(\mathbf{x}^{k+1},\mathbf{u}^{k+1})$, the primal correctors at the $(k+1)$-th iteration of the PC^{2}PM algorithm. Also letting $\mathbf{z}=(\mathbf{x}^*,\mathbf{u}^*)$, but keeping $\bar{\mathbf{z}}=(\mathbf{x}^k,\mathbf{u}^k)$, by (3.15c) in Lemma 3.3.1 and Lemma 2.2.1, we have that:

$$2\rho^{k+1}\Bigg[\mathcal{R}(\widehat{\mathbf{z}};\zeta^{k+1})-\mathcal{R}(\mathbf{z};\zeta^{k+1})\Bigg]\leq\|\bar{\mathbf{z}}-\mathbf{z}\|_2^2-\|\widehat{\mathbf{z}}-\mathbf{z}\|_2^2-\|\widehat{\mathbf{z}}-\bar{\mathbf{z}}\|_2^2,$$

which leads to the following expanded inequality

$$\begin{aligned}
&2\rho^{k+1}\Bigg\{(P_0\mathbf{y}^{k+1}+\mathbf{q}_0)^T\mathbf{x}^{k+1}+\mathbf{c}_0^T\mathbf{u}^{k+1}+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m_1}\mu_{\mathrm{i}}^{k+1}\Big[(P_{\mathrm{i}}\mathbf{y}^{k+1}+\mathbf{q}_{\mathrm{i}})^T\mathbf{x}^{k+1}+\mathbf{c}_{\mathrm{i}}^T\mathbf{u}^{k+1}+r_{\mathrm{i}}\Big]\\
&\qquad+(\boldsymbol{\nu}^{k+1})^T(A\mathbf{x}^{k+1}+B\mathbf{u}^{k+1}-\mathbf{b})\Bigg\}
\end{aligned}$$

$$
\begin{aligned}
&-2\rho^{k+1}\Bigg\{(P_0\mathbf{y}^{k+1}+\mathbf{q}_0)^T\mathbf{x}^*+\mathbf{c}_0^T\mathbf{u}^*+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m_1}\mu_\mathrm{i}^{k+1}\Big[(P_\mathrm{i}\mathbf{y}^{k+1}+\mathbf{q}_\mathrm{i})^T\mathbf{x}^*+\mathbf{c}_\mathrm{i}^T\mathbf{u}^*+r_\mathrm{i}\Big]\\
&\qquad+(\boldsymbol{\nu}^{k+1})^T(A\mathbf{x}^*+B\mathbf{u}^*-\mathbf{b})\Bigg\}\\
&\le\Big(\|\mathbf{x}^k-\mathbf{x}^*\|_2^2+\|\mathbf{u}^k-\mathbf{u}^*\|_2^2\Big)\\
&\quad-\Big(\|\mathbf{x}^{k+1}-\mathbf{x}^*\|_2^2+\|\mathbf{u}^{k+1}-\mathbf{u}^*\|_2^2\Big)-\Big(\|\mathbf{x}^{k+1}-\mathbf{x}^k\|_2^2+\|\mathbf{u}^{k+1}-\mathbf{u}^k\|_2^2\Big).
\end{aligned}\tag{3.40}
$$

The final piece to derive inequality (3.17) is to utilize Lemma 3.3.2. Let $(\mathbf{x}^*,\mathbf{u}^*,\boldsymbol{\lambda}^*,\boldsymbol{\gamma}^*)$ be a saddle point of QCQP (3.2), and again, $\zeta^{k+1}=(\mathbf{y}^{k+1},\mu^{k+1},\boldsymbol{\nu}^{k+1})$. By Lemma 3.3.2, we have that

$$
\begin{aligned}
&\mathcal{R}(\mathbf{x}^*,\mathbf{u}^*;\zeta^{k+1})-\mathcal{R}(\mathbf{y}^{k+1},\mathbf{v}^{k+1};\zeta^{k+1})\\
\le\ &\sum_{\mathrm{i}=1}^{m}\left[(\lambda_\mathrm{i}^*-\mu_\mathrm{i}^{k+1})\left(\frac{1}{2}\mathbf{y}^{k+1^T}P_\mathrm{i}\mathbf{y}^{k+1}+\mathbf{q}_\mathrm{i}^T\mathbf{y}^{k+1}+\mathbf{c}_\mathrm{i}^T\mathbf{v}^{k+1}+r_\mathrm{i}\right)\right]\\
&+(\boldsymbol{\gamma}^*-\boldsymbol{\nu}^{k+1})^T(A\mathbf{y}^{k+1}+B\mathbf{v}^{k+1}-\mathbf{b}).
\end{aligned}
$$

Multiplying both sides by $2\rho^{k+1}$ and expanding the $\mathcal{R}$ function, we have that

$$
\begin{aligned}
&2\rho^{k+1}\Bigg\{(P_0\mathbf{y}^{k+1}+\mathbf{q}_0)^T\mathbf{x}^*+\mathbf{c}_0^T\mathbf{u}^*+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m}\mu_\mathrm{i}^{k+1}\Big[(P_\mathrm{i}\mathbf{y}^{k+1}+\mathbf{q}_\mathrm{i})^T\mathbf{x}^*+\mathbf{c}_\mathrm{i}^T\mathbf{u}^*+r_\mathrm{i}\Big]\\
&\qquad+(\boldsymbol{\nu}^{k+1})^T(A\mathbf{x}^*+B\mathbf{u}^*-\mathbf{b})\Bigg\}\\
&-2\rho^{k+1}\Bigg\{(P_0\mathbf{y}^{k+1}+\mathbf{q}_0)^T\mathbf{y}^{k+1}+\mathbf{c}_0^T\mathbf{v}^{k+1}+r_0\\
&\qquad+\sum_{\mathrm{i}=1}^{m}\mu_\mathrm{i}^{k+1}\Big[(P_\mathrm{i}\mathbf{y}^{k+1}+\mathbf{q}_\mathrm{i})^T\mathbf{y}^{k+1}+\mathbf{c}_\mathrm{i}^T\mathbf{v}^{k+1}+r_\mathrm{i}\Big]\\
&\qquad+(\boldsymbol{\nu}^{k+1})^T(A\mathbf{y}^{k+1}+B\mathbf{v}^{k+1}-\mathbf{b})\Bigg\}
\end{aligned}
$$

$$\leq 2\rho^{k+1}\Bigg\{\sum_{\mathrm{i}=1}^{m}(\lambda_{\mathrm{i}}^{*}-\mu_{\mathrm{i}}^{k+1})\Big[\frac{1}{2}(\mathbf{y}^{k+1})^{T}P_{\mathrm{i}}\mathbf{y}^{k+1}+\mathbf{q}_{\mathrm{i}}^{T}\mathbf{y}^{k+1}+\mathbf{c}_{\mathrm{i}}^{T}\mathbf{v}^{k+1}+r_{\mathrm{i}}\Big]$$
$$+(\boldsymbol{\gamma}^{*}-\boldsymbol{\nu}^{k+1})^{T}(A\mathbf{y}^{k+1}+B\mathbf{v}^{k+1}-\mathbf{b})\Bigg\}. \tag{3.41}$$

Adding the three inequalities (3.39), (3.40) and (3.41) yields the inequality (3.17) in Proposition 3.3.3.

To prove the second inequality, (3.18), in Proposition 3.3.3, we use a similar approach as above, just replacing the linear approximation function $\mathcal{R}$ with the original Lagrangian function $\mathcal{L}$. More specifically, let $\widehat{\mathbf{z}}=(\boldsymbol{\mu}^{k+1},\boldsymbol{\nu}^{k+1})$. By (3.15a) in Lemma 3.3.1, we know that

$$\begin{aligned}\widehat{\mathbf{z}}:=&\ (\boldsymbol{\mu}^{k+1},\ \boldsymbol{\nu}^{k+1})\\ =&\ \underset{\boldsymbol{\lambda}\in\mathbb{R}_{+}^{m_1},\ \boldsymbol{\gamma}\in\mathbb{R}^{m_2}}{\operatorname{argmin}}\ -\mathcal{L}(\mathbf{x}^{k},\mathbf{u}^{k},\boldsymbol{\lambda},\boldsymbol{\gamma})+\tfrac{1}{2\rho^{k+1}}\|\boldsymbol{\lambda}-\boldsymbol{\lambda}^{k}\|_2^2+\tfrac{1}{2\rho^{k+1}}\|\boldsymbol{\gamma}-\boldsymbol{\gamma}^{k}\|_2^2.\end{aligned}$$

Letting $\bar{\mathbf{z}}=(\boldsymbol{\lambda}^{k},\boldsymbol{\gamma}^{k})$ and choosing a specific $\mathbf{z}=(\boldsymbol{\lambda}^{k+1},\boldsymbol{\gamma}^{k+1})$, we use Lemma 2.2.1 to obtain that

$$2\rho^{k+1}\Bigg[\Big(-\mathcal{L}(\mathbf{x}^{k},\mathbf{u}^{k};\widehat{\mathbf{z}})\Big)-\Big(-\mathcal{L}(\mathbf{x}^{k},\mathbf{u}^{k};\mathbf{z})\Big)\Bigg]\leq\|\bar{\mathbf{z}}-\mathbf{z}\|_2^2-\|\widehat{\mathbf{z}}-\mathbf{z}\|_2^2-\|\widehat{\mathbf{z}}-\bar{\mathbf{z}}\|_2^2,$$

which yields the following expanded inequality:

$$\begin{aligned}&2\rho^{k+1}\Bigg\{\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^{k+1}-\mu_{\mathrm{i}}^{k+1})\Big[\frac{1}{2}(\mathbf{x}^{k})^{T}P_{\mathrm{i}}\mathbf{x}^{k}+\mathbf{q}_0^{T}\mathbf{x}^{k}+\mathbf{c}_0^{T}\mathbf{u}^{k}+r_{\mathrm{i}}\Big]\\&\qquad+(\boldsymbol{\gamma}^{k+1}-\boldsymbol{\nu}^{k+1})^{T}(A\mathbf{x}^{k}+B\mathbf{u}^{k}-\mathbf{b})\Bigg\}\\&\leq\Big(\|\boldsymbol{\lambda}^{k}-\boldsymbol{\lambda}^{k+1}\|_2^2+\|\boldsymbol{\gamma}^{k}-\boldsymbol{\gamma}^{k+1}\|_2^2\Big)\\&-\Big(\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1}\|_2^2\Big)-\Big(\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2+\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2\Big).\end{aligned}\tag{3.42}$$

Similarly, again with some abuse of notation, letting $\widehat{\mathbf{z}} = (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})$, by (3.15d) in Lemma 3.3.1, we have that

$$\begin{aligned} \widehat{\mathbf{z}} := \;& (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1}) \\ = \;& \underset{\boldsymbol{\lambda}\in\mathbb{R}^{m_1}_{+},\ \boldsymbol{\gamma}\in\mathbb{R}^{m_2}}{\operatorname{argmin}} - \mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) + \tfrac{1}{2\rho^{k+1}}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^k\|_2^2 + \tfrac{1}{2\rho^{k+1}}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^k\|_2^2. \end{aligned}$$

By choosing $\mathbf{z}$ to be $(\boldsymbol{\lambda}^*, \boldsymbol{\gamma}^*)$, while keeping $\bar{\mathbf{z}}$ at $(\boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$, we have from Lemma 2.2.1 that

$$\begin{aligned} & 2\rho^{k+1}\left[\Big(- \mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}; \widehat{\mathbf{z}})\Big) - \Big(- \mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}; \mathbf{z})\Big)\right] \\ \leq \;& \|\bar{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\widehat{\mathbf{z}} - \mathbf{z}\|_2^2 - \|\widehat{\mathbf{z}} - \bar{\mathbf{z}}\|_2^2, \end{aligned}$$

which yields the following expanded inequality:

$$\begin{aligned} & 2\rho^{k+1}\Bigg\{\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^* - \lambda_{\mathrm{i}}^{k+1})\left[\frac{1}{2}(\mathbf{y}^{k+1})^T P_{\mathrm{i}}\mathbf{y}^{k+1} + \mathbf{q}_0^T\mathbf{y}^{k+1} + \mathbf{c}_0^T\mathbf{v}^{k+1} + r_{\mathrm{i}}\right] \\ & \qquad + (\boldsymbol{\gamma}^* - \boldsymbol{\gamma}^{k+1})^T(A\mathbf{y}^{k+1} + B\mathbf{v}^{k+1} - \mathbf{b})\Bigg\} \\ & \leq \Big(\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^*\|_2^2\Big) \\ & - \Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^*\|_2^2\Big) - \Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^k\|_2^2\Big). \end{aligned} \tag{3.43}$$

Adding the two inequalities (3.42) and (3.43) leads to the second inequality, (3.18), in Proposition 3.3.3.

□

3.7.2 Proof of Theorem 3.3.4

By adding the two inequalities (3.17) and (3.18) in Proposition 3.3.3, we have that

$$\begin{aligned} & \|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^*\|_2^2 \\ \leq \;& \|\mathbf{x}^k - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^k - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^*\|_2^2 \\ - \;& \Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big) \end{aligned}$$

$$
\begin{aligned}
&- \Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k+1}\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k}\|_2^2 \Big) \\
&+ \underbrace{2\rho^{k+1}(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_0 (\mathbf{y}^{k+1} - \mathbf{x}^{k})}_{\text{(a)}} \\
&+ \sum_{\mathrm{i}=1}^{m_1} \underbrace{2\rho^{k+1}\mu_{\mathrm{i}}^{k+1}(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_{\mathrm{i}} (\mathbf{y}^{k+1} - \mathbf{x}^{k})}_{\text{(b)}_{\mathrm{i}}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^{k+1} - \mu_{\mathrm{i}}^{k+1})\Big[\frac{1}{2}(\mathbf{y}^{k+1})^T P_{\mathrm{i}} \mathbf{y}^{k+1} - \frac{1}{2}(\mathbf{x}^{k})^T P_{\mathrm{i}} \mathbf{x}^{k}\Big]}_{\text{(c)}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^{k+1} - \mu_{\mathrm{i}}^{k+1})\mathbf{q}_{\mathrm{i}}^T(\mathbf{y}^{k+1} - \mathbf{x}^{k})}_{\text{(d)}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\mu_{\mathrm{i}}^{k+1} - \lambda_{\mathrm{i}}^{k})\mathbf{q}_{\mathrm{i}}^T(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})}_{\text{(e)}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\mu_{\mathrm{i}}^{k+1} - \lambda_{\mathrm{i}}^{k})(P_{\mathrm{i}}\mathbf{x}^{k})^T(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})}_{\text{(f)}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\lambda_{\mathrm{i}}^{k+1} - \mu_{\mathrm{i}}^{k+1})\mathbf{c}_{\mathrm{i}}^T(\mathbf{v}^{k+1} - \mathbf{u}^{k})}_{\text{(g)}} \\
&+ \underbrace{2\rho^{k+1}\sum_{\mathrm{i}=1}^{m_1}(\mu_{\mathrm{i}}^{k+1} - \lambda_{\mathrm{i}}^{k})\mathbf{c}_{\mathrm{i}}^T(\mathbf{v}^{k+1} - \mathbf{u}^{k+1})}_{\text{(h)}} \\
&+ \underbrace{2\rho^{k+1}(\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1})^T A(\mathbf{y}^{k+1} - \mathbf{x}^{k})}_{\text{(i)}} + \underbrace{2\rho^{k+1}(\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k})^T A(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})}_{\text{(j)}} \\
&+ \underbrace{2\rho^{k+1}(\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1})^T B(\mathbf{v}^{k+1} - \mathbf{u}^{k})}_{\text{(k)}} + \underbrace{2\rho^{k+1}(\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k})^T B(\mathbf{v}^{k+1} - \mathbf{u}^{k+1})}_{\text{(l)}} \qquad (3.44)
\end{aligned}
$$

Next, we establish an upper bound for each term of the term from (a) to (l) in (3.44) using the adaptive step size $\rho^{k+1} = \rho(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$, as defined in (3.19).

(a) First, we want to show that

$$
\text{(a)} \leq \epsilon_1 \Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k}\|_2^2 \Big). \qquad (3.45)
$$

To prove this (and several inequalities below), we first restate the Young's inequality: given any two vectors $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^n$, we have

$$\mathbf{z}_1^T \mathbf{z}_2 \leq \sum_{\mathrm{j}=1}^{n} \left[\frac{1}{2}\Big(\frac{1}{\delta} z_{1\mathrm{j}}\Big)^2 + \frac{1}{2}\Big(\delta z_{2\mathrm{j}}\Big)^2 \right] = \frac{1}{2\delta^2}\|\mathbf{z}_1\|_2^2 + \frac{\delta^2}{2}\|\mathbf{z}_2\|_2^2. \tag{3.46}$$

Applying (3.46) on (a) yields

$$\begin{aligned}
\text{(a)} \leq& 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|P_0(\mathbf{y}^{k+1} - \mathbf{x}^k)\|_2^2\Big) \\
\leq& 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \frac{\delta^2}{2}|||P_0|||_2^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big) \\
\leq& 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|P_0\|_F^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big).
\end{aligned} \tag{3.47}$$

The second inequality holds due to the property that given a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $\mathbf{z} \in \mathbb{R}^n$, $\|A\mathbf{z}\|_2 \leq |||A|||_2\|\mathbf{z}\|_2$ (see Theorem 5.6.2 in [38]), where we use the notation $|||\cdot|||_2$ to denote the matrix norm $|||A|||_2 \coloneqq \sup\limits_{\mathbf{z}\neq\mathbf{0}} \frac{\|A\mathbf{z}\|_2}{\|\mathbf{z}\|_2}$. The last inequality holds due to the property $|||A|||_2 \leq \|A\|_F$ [39], where $\|A\|_F \coloneqq \Big(\sum_{\mathrm{i}=1}^m \sum_{\mathrm{j}=1}^n |A_{\mathrm{ij}}|^2\Big)^{\frac{1}{2}}$ denotes the Frobenius norm.

From (3.47), if $\|P_0\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|P_0\|_F}$ yields

$$\text{(a)} \leq \rho^{k+1}\|P_0\|_F\Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.48}$$

Since $\rho^{k+1} \leq \rho_1 = \frac{\epsilon_1}{\|P_0\|_F}$, we obtain (3.45). If, on the other hand, $\|P_0\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(a)} \leq \rho^{k+1}\Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.49}$$

Since $\rho^{k+1} \leq \rho_1 = \epsilon_1$, (3.45) is also obtained.

(b) Here we want to show that

$$\sum_{\mathrm{i}=1}^{m_1} \text{(b)}_{\mathrm{i}} \leq \epsilon_2\Big(\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.50}$$

Applying (3.46) on each term $(\mathrm{b})_{\mathrm{i}}$ yields

$$\begin{aligned}(\mathrm{b})_{\mathrm{i}} \leq& 2\rho^{k+1}\mu_{\mathrm{i}}^{k+1}\Big(\frac{1}{2\delta_{\mathrm{i}}^2}\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\frac{\delta_{\mathrm{i}}^2}{2}\|P_{\mathrm{i}}\|_2^2\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big)\\ \leq& 2\rho^{k+1}\mu_{\mathrm{i}}^{k+1}\Big(\frac{1}{2\delta_{\mathrm{i}}^2}\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\frac{\delta_{\mathrm{i}}^2}{2}\|P_{\mathrm{i}}\|_F^2\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big).\end{aligned} \tag{3.51}$$

- If $\|P_{\mathrm{i}}\|_F \neq 0$, then letting $\delta_{\mathrm{i}}^2=\frac{1}{\|P_{\mathrm{i}}\|_F}$ yields

$$\begin{aligned}(\mathrm{b})_{\mathrm{i}} \leq& \rho^{k+1}\mu_{\mathrm{i}}^{k+1}\|P_{\mathrm{i}}\|_F\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big)\\ \leq& \rho^{k+1}\tilde{\mu}_{\mathrm{i}}^{k+1}\|P_{\mathrm{i}}\|_F\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big),\end{aligned} \tag{3.52}$$

where $\tilde{\mu}_{\mathrm{i}}^{k+1} := \lambda_{\mathrm{i}}^k+\rho^{k+1}|\frac{1}{2}(\mathbf{x}^k)^TP_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}}^T\mathbf{x}^k+\mathbf{c}_{\mathrm{i}}^T\mathbf{u}^k+r_{\mathrm{i}}| \geq \mu_{\mathrm{i}}^{k+1}$. If we can bound $\rho^{k+1}\tilde{\mu}_{\mathrm{i}}^{k+1} \leq \frac{\epsilon_2}{m_1\|P_{\mathrm{i}}\|_F}$, then we can achieve

$$(\mathrm{b})_{\mathrm{i}} \leq \frac{\epsilon_2}{m_1}\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big). \tag{3.53}$$

By substituting $a_{\mathrm{i}}=|\frac{1}{2}(\mathbf{x}^k)^TP_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}}^T\mathbf{x}^k+\mathbf{c}_{\mathrm{i}}^T\mathbf{u}^k+r_{\mathrm{i}}| \geq 0$, $b_{\mathrm{i}}=\lambda_{\mathrm{i}}^k \geq 0$ and $c_{\mathrm{i}}=\frac{\epsilon_2}{m_1\|P_{\mathrm{i}}\|_F} > 0$, we can rewrite $\rho^{k+1}\tilde{\mu}_{\mathrm{i}}^{k+1}-\frac{\epsilon_2}{m_1\|P_{\mathrm{i}}\|_F}$ as $a_{\mathrm{i}}(\rho^{k+1})^2+b_{\mathrm{i}}\rho^{k+1}-c_{\mathrm{i}}$, which is simply a quadratic function of ρ^{k+1} with parameters a_{i}, b_{i} and c_{i}. To bound $\rho^{k+1}\tilde{\mu}_{\mathrm{i}}^{k+1} \leq \frac{\epsilon_2}{m_1\|P_{\mathrm{i}}\|_F}$ is equivalent to find proper values of ρ^{k+1} that keep the quadratic function stay below zero.

  – If $a_{\mathrm{i}}=0$ and $b_{\mathrm{i}}=0$, then $\rho^{k+1}\in(0,+\infty)$.
  – If $a_{\mathrm{i}}=0$ and $b_{\mathrm{i}}>0$, then $\rho^{k+1}\in(0,\frac{c_{\mathrm{i}}}{b_{\mathrm{i}}}]$.
  – If $a_{\mathrm{i}}>0$, then $\rho^{k+1}\in(0,\frac{-b_{\mathrm{i}}+\sqrt{b_{\mathrm{i}}^2+4a_{\mathrm{i}}c_{\mathrm{i}}}}{2a_{\mathrm{i}}}]$.

  Since $\rho^{k+1}\leq\rho_2(\mathbf{x}^k,\mathbf{u}^k,\boldsymbol{\lambda}^k)\leq\rho_{2\mathrm{i}}(\mathbf{x}^k,\mathbf{u}^k,\boldsymbol{\lambda}^k)$, it satisfies all the above three conditions, we then obtain (3.53), and hence (3.50).

- If $\|P_{\mathrm{i}}\|_F=0$, then letting $\delta_{\mathrm{i}}^2=1$ yields

$$\begin{aligned}(\mathrm{b})_{\mathrm{i}} \leq& \rho^{k+1}\mu_{\mathrm{i}}^{k+1}\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big)\\ \leq& \rho^{k+1}\tilde{\mu}_{\mathrm{i}}^{k+1}\Big(\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\Big).\end{aligned} \tag{3.54}$$

Similarly, if we can bound $\rho^{k+1}\tilde{\mu}_{\rm i}^{k+1} \leq \frac{\epsilon_2}{m_1}$, then we can also achieve (3.53). By substituting $a_{\rm i} = |\frac{1}{2}(\mathbf{x}^k)^T P_{\rm i}\mathbf{x}^k + \mathbf{q}_{\rm i}^T\mathbf{x}^k + \mathbf{c}_{\rm i}^T\mathbf{u}^k + r_{\rm i}| \geq 0$, $b_{\rm i} = \lambda_{\rm i}^k \geq 0$ and $c_{\rm i} = \frac{\epsilon_2}{m_1} > 0$, we can rewrite $\rho^{k+1}\tilde{\mu}_{\rm i}^{k+1} - \frac{\epsilon_2}{m_1}$ as $a_{\rm i}(\rho^{k+1})^2 + b_{\rm i}\rho^{k+1} - c_{\rm i}$. The same analysis can be followed as discussed in the case of $\|P_{\rm i}\|_F \neq 0$.

(c) Next, we want to show that

$$\text{(c)} \leq \epsilon_3\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.55}$$

By using P to denote $\begin{pmatrix} P_1 \\ \vdots \\ P_{m_1} \end{pmatrix}$, we can rewrite

$$\text{(c)} = \rho^{k+1}\Big\{(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1})^T\Big[I_{m_1\times m_1} \otimes (\mathbf{x}^k + \mathbf{y}^{k+1})^T\Big]P(\mathbf{y}^{k+1} - \mathbf{x}^k)\Big\}, \tag{3.56}$$

where $\otimes$ denotes the Kronecker product; that is, given a matrix $A \in \mathbb{R}^{m_1\times n_1}$ and a matrix $B \in \mathbb{R}^{m_2\times n_2}$, $A \otimes B := \begin{pmatrix} a_{11}B & \cdots & a_{1m_1}B \\ \vdots & & \vdots \\ a_{m_1 1}B & \cdots & a_{m_1 n_1}B \end{pmatrix}$. Applying (3.46) to (3.56) yields

$$\begin{aligned} \text{(c)} \leq \rho^{k+1}\Big(&\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 \\ &+ \frac{\delta^2}{2}\Big\|\Big|I_{m_1\times m_1} \otimes (\mathbf{x}^k + \mathbf{y}^{k+1})^T\Big\|\Big|_2^2 \||P\||_2^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big) \\ \leq \rho^{k+1}\Big(&\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|\mathbf{x}^k + \mathbf{y}^{k+1}\|_2^2\|P\|_F^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \end{aligned} \tag{3.57}$$

Since we have the property that $|||A \otimes B|||_2 = |||A|||_2|||B|||_2$ (see Theorem 8 in [40]), the last inequality holds due to

$$\Big|\Big|\Big|I_{m_1\times m_1} \otimes (\mathbf{x}^k + \mathbf{y}^{k+1})^T\Big|\Big|\Big|_2^2 = |||I_{m_1\times m_1}|||_2^2\Big|\Big|\Big|(\mathbf{x}^k + \mathbf{y}^{k+1})^T\Big|\Big|\Big|_2^2,$$

together with $|||I_{m_1\times m_1}|||_2 = 1$ and $\Big|\Big|\Big|(\mathbf{x}^k + \mathbf{y}^{k+1})^T\Big|\Big|\Big|_2 \leq \|(\mathbf{x}^k + \mathbf{y}^{k+1})^T\|_F = \|\mathbf{x}^k + \mathbf{y}^{k+1}\|_2$. Note that $\|P\|_F \neq 0$, otherwise the QCQP is simply a QP.

- If $\|\mathbf{x}^k + \mathbf{y}^{k+1}\|_2 \neq 0$, then letting $\delta^2 = \frac{1}{\|\mathbf{x}^k+\mathbf{y}^{k+1}\|_2\|P\|_F}$ yields

$$\begin{aligned}\text{(c)} \leq & \frac{1}{2}\rho^{k+1}\|\mathbf{x}^k + \mathbf{y}^{k+1}\|_2\|P\|_F\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big) \\ \leq & \frac{1}{2}\rho^{k+1}\|\mathbf{x}^k + \tilde{\mathbf{y}}^{k+1}\|_2\|P\|_F\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big),\end{aligned} \tag{3.58}$$

where $\tilde{y}_{\mathrm{j}}^{k+1} := x_{\mathrm{j}}^k + \rho\Big|\Big[P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\Big(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\gamma}^k\Big]_{\mathrm{j}}\Big| \geq y_{\mathrm{j}}^{k+1}$. If we can bound $\rho^{k+1}\|\mathbf{x}^k + \tilde{\mathbf{y}}^{k+1}\|_2 \leq \frac{2\epsilon_3}{\|P^T\|_F}$, then (3.55) can be obtained. We first bound

$$\begin{aligned}&\rho^{k+1}\|\mathbf{x}^k + \tilde{\mathbf{y}}^{k+1}\|_2 - \frac{2\epsilon_3}{\|P\|_F} \\ \leq&\rho^{k+1}\Big[2\|\mathbf{x}^k\|_2 + \rho^{k+1}\|P_0\mathbf{x}^k + \mathbf{q}_0 + \sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\Big(P_{\mathrm{i}}\mathbf{x}^k + \mathbf{q}_{\mathrm{i}}\Big) + A^T\boldsymbol{\gamma}^k\|_2\Big] \\ &-\frac{2\epsilon_3}{\|P\|_F}.\end{aligned} \tag{3.59}$$

By substituting $a = \|P_0\mathbf{x}^k+\mathbf{q}_0+\sum_{\mathrm{i}=1}^{m_1}\lambda_{\mathrm{i}}^k\Big(P_{\mathrm{i}}\mathbf{x}^k+\mathbf{q}_{\mathrm{i}}\Big)+A^T\boldsymbol{\gamma}^k\|_2 \geq 0$, $b = 2\|\mathbf{x}^k\|_2 \geq 0$ and $c = \frac{2\epsilon_3}{\|P\|_F} > 0$, we can bound $\rho^{k+1}\|\mathbf{x}^k+\tilde{\mathbf{y}}^{k+1}\|_2-\frac{2\epsilon_3}{\|P\|_F}$ using $a(\rho^{k+1})^2+b\rho^{k+1}-c$, which is simply a quadratic function of ρ^{k+1} with parameters a, b and c. Bounding $\rho^{k+1}\|\mathbf{x}^k + \tilde{\mathbf{y}}^{k+1}\|_2 \leq \frac{2\epsilon_3}{\|P\|_F}$ can be guaranteed by finding the proper values of ρ^{k+1} that keep the quadratic function stay below zero.

  - If $a = 0$ and $b = 0$, then $\rho^{k+1} \in (0, +\infty)$.
  - If $a = 0$ and $b > 0$, then $\rho^{k+1} \in (0, \frac{c}{b}]$.
  - If $a > 0$, then $\rho^{k+1} \in (0, \frac{-b+\sqrt{b^2+4ac}}{2a}]$.

Since $\rho^{k+1} \leq \rho_3(\mathbf{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$, it satisfies all the above three conditions, we obtain (3.55).

- If $\|\mathbf{x}^k + \mathbf{y}^{k+1}\|_2 = 0$, then letting $\delta^2 = 1$ yields

$$(c) \leq \frac{1}{2}\rho^{k+1}\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big) \tag{3.60}$$

Since $\rho^{k+1} \leq \rho_3(\mathbf{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k) \leq 2\epsilon_3$, (3.55) is also obtained.

(d) To show that

$$\text{(d)} \leq \epsilon_4\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big), \tag{3.61}$$

by letting $Q = \begin{pmatrix} \mathbf{q}_1^T \\ \vdots \\ \mathbf{q}_{m_1}^T \end{pmatrix}$, we can rewrite that

$$\text{(d)} = 2\rho^{k+1}\Big[(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1})^T Q^T (\mathbf{y}^{k+1} - \mathbf{x}^k)\Big]. \tag{3.62}$$

Applying (3.46) to (3.62) yields

$$\begin{aligned} \text{(d)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|\|Q\|\|_2^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big) \\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|Q\|_F^2\|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \end{aligned} \tag{3.63}$$

- If $\|Q\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|Q\|_F}$ yields

$$\text{(d)} \leq \rho^{k+1}\|Q\|_F\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.64}$$

  Since $\rho^{k+1} \leq \rho_4 = \frac{\epsilon_4}{\|Q\|_F}$, we obtain (3.61).

- If $\|Q\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(d)} \leq \rho^{k+1}\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big). \tag{3.65}$$

  Since $\rho^{k+1} \leq \rho_4 = \epsilon_4$, (3.61) is also obtained.

(e) Similarly, to show

$$\text{(e)} \leq \epsilon_4\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big), \tag{3.66}$$

we can rewrite that

$$\text{(e)} = 2\rho^{k+1}\Big[(\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k)^T Q (\mathbf{y}^{k+1} - \mathbf{x}^{k+1})\Big]. \tag{3.67}$$

Applying (3.46) to (3.67) yields that

$$\begin{aligned}\text{(e)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{\delta^2}{2}\|Q\|_2^2\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big) \\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{\delta^2}{2}\|Q\|_F^2\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big).\end{aligned} \tag{3.68}$$

- If $\|Q\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|Q\|_F}$ yields

$$\text{(e)} \leq \rho^{k+1}\|Q\|_F\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big). \tag{3.69}$$

  Since $\rho^{k+1} \leq \rho_4 = \frac{\epsilon_4}{\|Q\|_F}$, we obtain (3.66).

- If $\|Q\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(e)} \leq \rho^{k+1}\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big). \tag{3.70}$$

  Since $\rho^{k+1} \leq \rho_4 = \epsilon_4$, (3.66) is also obtained.

(f) To show

$$\text{(f)} \leq \epsilon_5\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big), \tag{3.71}$$

we can rewrite

$$\text{(f)} = 2\rho^{k+1}\Big\{(\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k)^T\Big[I_{m_1\times m_1} \otimes (\mathbf{x}^k)^T\Big]P(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})\Big\}. \tag{3.72}$$

Applying (3.46), we have that

$$\begin{aligned}\text{(f)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 \\ & + \frac{\delta^2}{2}\Big\|I_{m_1\times m_1} \otimes (\mathbf{x}^k)^T\Big\|_2^2\|P\|_2^2\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big) \\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{\delta^2}{2}\|\mathbf{x}^k\|_2^2\|P\|_F^2\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big).\end{aligned} \tag{3.73}$$

Similarly, the last inequality holds due to

$$\Big\|I_{m_1\times m_1} \otimes (\mathbf{x}^k)^T\Big\|_2^2 = \|I_{m_1\times m_1}\|_2^2\Big\|(\mathbf{x}^k)^T\Big\|_2^2.$$

- If $\|\mathbf{x}^k\|_2 \neq 0$, then letting $\delta^2 = \frac{1}{\|\mathbf{x}^k\|_2\|P\|_F}$ yields

$$(\text{f}) \leq \rho^{k+1}\|\mathbf{x}^k\|_2\|P\|_F\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big). \tag{3.74}$$

  Since $\rho^{k+1} \leq \rho_5(\mathbf{x}^k) = \frac{\epsilon_5}{\|\mathbf{x}^k\|_2\|P\|_F}$, we obtain (3.71).

- If $\|\mathbf{x}^k\|_2 = 0$, then letting $\delta^2 = 1$ yields

$$(\text{f}) \leq \rho^{k+1}\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big). \tag{3.75}$$

  Since $\rho^{k+1} \leq \rho_5(\mathbf{x}^k) = \epsilon_5$, (3.71) is also obtained.

(g) To show

$$(\text{g}) \leq \epsilon_6\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big), \tag{3.76}$$

By letting $C = \begin{pmatrix} \mathbf{c}_1^T \\ \vdots \\ \mathbf{c}_{m_2}^T \end{pmatrix}$, we can rewrite

$$(\text{g}) = 2\rho^{k+1}\Big[(\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1})^T C(\mathbf{v}^{k+1} - \mathbf{u}^k)\Big]. \tag{3.77}$$

Applying (3.46), we have that

$$\begin{aligned}(\text{g}) \leq& 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|C\|_2^2\|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big) \\ \leq& 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|C\|_F^2\|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big).\end{aligned} \tag{3.78}$$

- If $\|C\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|C\|_F}$ yields

$$(\text{g}) \leq \rho^{k+1}\|C\|_F\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big). \tag{3.79}$$

  Since $\rho^{k+1} \leq \rho_6 = \frac{\epsilon_6}{\|C\|_F}$, we obtain (3.76).

- If $\|C\|_F = 0$, then letting $\delta^2 = 1$ yields

$$(\mathrm{g}) \leq \rho^{k+1}\Big(\|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\mu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big). \tag{3.80}$$

  Since $\rho^{k+1} \leq \rho_6 = \epsilon_6$, (3.76) is also obtained.

(h) Next, we want to show that

$$(\mathrm{h}) \leq \epsilon_6\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big). \tag{3.81}$$

Similarly, we can rewrite

$$(\mathrm{h}) = 2\rho^{k+1}\Big[(\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k)^T C(\mathbf{v}^{k+1} - \mathbf{u}^{k+1})\Big]. \tag{3.82}$$

Applying (3.46) on the above equality leads to

$$\begin{aligned}(\mathrm{h}) \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{\delta^2}{2}\|C\|_2^2\|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big)\\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \frac{\delta^2}{2}\|C\|_F^2\|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big).\end{aligned} \tag{3.83}$$

- If $\|C\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|C\|_F}$ yields

$$(\mathrm{h}) \leq \rho^{k+1}\|C\|_F\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big). \tag{3.84}$$

  Since $\rho^{k+1} \leq \rho_6 = \frac{\epsilon_6}{\|C\|_F}$, we obtain (3.81).

- If $\|C\|_F = 0$, then letting $\delta^2 = 1$ yields

$$(\mathrm{h}) \leq \rho^{k+1}\Big(\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big). \tag{3.85}$$

  Since $\rho^{k+1} \leq \rho_6 = \epsilon_6$, (3.81) is also obtained.

(i) To show

$$(\mathrm{i}) \leq \epsilon_7\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^k\|_2^2\Big), \tag{3.86}$$

we apply (3.46) on the rewriting of (i), which leads to

$$\begin{aligned}\text{(i)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\nu}^{k+1}\|_2^2+\frac{\delta^2}{2}\|\|A\|_2^2\|\mathbf{y}^{k+1}-\mathbf{x}^k\|_2^2\Big)\\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\nu}^{k+1}\|_2^2+\frac{\delta^2}{2}\|A\|_F^2\|\mathbf{y}^{k+1}-\mathbf{x}^k\|_2^2\Big).\end{aligned} \tag{3.87}$$

- If $\|A\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|A\|_F}$ yields

$$\text{(i)} \leq \rho^{k+1}\|A\|_F\Big(\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\nu}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^k\|_2^2\Big). \tag{3.88}$$

  Since $\rho^{k+1} \leq \rho_7 = \frac{\epsilon_7}{\|A\|_F}$, we obtain (3.86).

- If $\|A\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(i)} \leq \rho^{k+1}\Big(\|\boldsymbol{\gamma}^{k+1}-\boldsymbol{\nu}^{k+1}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^k\|_2^2\Big). \tag{3.89}$$

  Since $\rho^{k+1} \leq \rho_7 = \epsilon_7$, (3.86) is also obtained.

(j) Similarly, to show

$$\text{(j)} \leq \epsilon_7\Big(\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2\Big). \tag{3.90}$$

we apply (3.46) on the rewriting of (j), which yields

$$\begin{aligned}\text{(j)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2+\frac{\delta^2}{2}\|\|A\|_2^2\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2\Big)\\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2+\frac{\delta^2}{2}\|A\|_F^2\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2\Big).\end{aligned} \tag{3.91}$$

- If $\|A\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|A\|_F}$ yields

$$\text{(j)} \leq \rho^{k+1}\|A\|_F\Big(\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2+\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2\Big). \tag{3.92}$$

  Since $\rho^{k+1} \leq \rho_7 = \frac{\epsilon_7}{\|A\|_F}$, we obtain (3.90).

- If $\|A\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(j)} \leq \rho^{k+1}\Big(\|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2\Big). \tag{3.93}$$

  Since $\rho^{k+1} \leq \rho_7 = \epsilon_7$, (3.90) is also obtained.

(k) Next, to show

$$\text{(k)} \leq \epsilon_8\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big), \tag{3.94}$$

we apply (3.46) on the rewriting of (k):

$$\begin{aligned} \text{(k)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|\|B\|\|_2^2\|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big) \\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \frac{\delta^2}{2}\|B\|_F^2\|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big). \end{aligned} \tag{3.95}$$

- If $\|B\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|B\|_F}$ yields

$$\text{(k)} \leq \rho^{k+1}\|B\|_F\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big). \tag{3.96}$$

  Since $\rho^{k+1} \leq \rho_8 = \frac{\epsilon_8}{\|B\|_F}$, we obtain (3.94).

- If $\|B\|_F = 0$, then letting $\delta^2 = 1$ yields

$$\text{(k)} \leq \rho^{k+1}\Big(\|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\nu}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^k\|_2^2\Big). \tag{3.97}$$

  Since $\rho^{k+1} \leq \rho_8 = \epsilon_8$, (3.94) is also obtained.

(l) Last, to show

$$\text{(l)} \leq \epsilon_8\Big(\|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big), \tag{3.98}$$

we apply (3.46) on the rewriting of (l):

$$\begin{aligned} \text{(l)} \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 + \frac{\delta^2}{2}\|\|B\|\|_2^2\|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big) \\ \leq & 2\rho^{k+1}\Big(\frac{1}{2\delta^2}\|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 + \frac{\delta^2}{2}\|B\|_F^2\|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2\Big). \end{aligned} \tag{3.99}$$

- If $\|B\|_F \neq 0$, then letting $\delta^2 = \frac{1}{\|B\|_F}$ yields

$$(\mathrm{l}) \leq \rho^{k+1}\|B\|_F\Big(\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^k\|_2^2 + \|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2\Big). \tag{3.100}$$

  Since $\rho^{k+1} \leq \rho_8 = \frac{\epsilon_8}{\|B\|_F}$, we obtain (3.98).

- If $\|B\|_F = 0$, then letting $\delta^2 = 1$ yields

$$(\mathrm{l}) \leq \rho^{k+1}\Big(\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^k\|_2^2 + \|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2\Big). \tag{3.101}$$

  Since $\rho^{k+1} \leq \rho_8 = \epsilon_8$, (3.98) is also obtained.

The summation of terms (a) to (l) can now be bounded as:

$$\begin{aligned}
&(\mathrm{a}) + \sum_{\mathrm{i}=1}^{m_1}(\mathrm{b})_{\mathrm{i}} + (\mathrm{c}) + (\mathrm{d}) + (\mathrm{e}) + (\mathrm{f}) + (\mathrm{g}) + (\mathrm{h}) + (\mathrm{i}) + (\mathrm{j}) + (\mathrm{k}) + (\mathrm{l})\\
\leq&(\epsilon_1+\epsilon_2+\epsilon_4+\epsilon_5+\epsilon_7)\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2\\
+&(\epsilon_1+\epsilon_2+\epsilon_3+\epsilon_4+\epsilon_7)\|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\\
+&(\epsilon_6+\epsilon_8)\|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2 + (\epsilon_6+\epsilon_8)\|\mathbf{v}^{k+1}-\mathbf{u}^{k}\|_2^2\\
+&(\epsilon_3+\epsilon_4+\epsilon_6)\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2 + (\epsilon_4+\epsilon_5+\epsilon_6)\|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2\\
+&(\epsilon_7+\epsilon_8)\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1}\|_2^2 + (\epsilon_7+\epsilon_8)\|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2\\
\leq&(\sum_{s=1}^{8}\epsilon_s)\Big[\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\\
&\qquad + \|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1}-\mathbf{u}^{k}\|_2^2\\
&\qquad + \|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2\\
&\qquad + \|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2\Big]\\
\leq&(1-\epsilon_0)\Big[\|\mathbf{y}^{k+1}-\mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1}-\mathbf{x}^{k}\|_2^2\\
&\qquad + \|\mathbf{v}^{k+1}-\mathbf{u}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1}-\mathbf{u}^{k}\|_2^2\\
&\qquad + \|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\mu}^{k+1}-\boldsymbol{\lambda}^{k}\|_2^2\\
&\qquad + \|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1}-\boldsymbol{\gamma}^{k}\|_2^2\Big].
\end{aligned} \tag{3.102}$$

Substituting it back into (3.44), we have that for all $k \geq 0$,

$$\begin{aligned}
&\|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^*\|_2^2 \\
\leq &\|\mathbf{x}^{k} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k} - \boldsymbol{\gamma}^*\|_2^2 \\
-&\epsilon_0 \Big[\|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{y}^{k+1} - \mathbf{x}^{k}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2 + \|\mathbf{v}^{k+1} - \mathbf{u}^{k}\|_2^2 \\
+&\|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 + \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k+1}\|_2^2 + \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k}\|_2^2 \Big],
\end{aligned} \tag{3.103}$$

which implies for all $k \geq 0$:

$$\begin{aligned}
0 \leq &\|\mathbf{x}^{k+1} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^*\|_2^2 \\
\leq &\|\mathbf{x}^{k} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k} - \boldsymbol{\gamma}^*\|_2^2 \\
\leq &\|\mathbf{x}^{k-1} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{k-1} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{k-1} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{k-1} - \boldsymbol{\gamma}^*\|_2^2 \\
\leq &\cdots \leq \|\mathbf{x}^{0} - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^{0} - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^{0} - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^{0} - \boldsymbol{\gamma}^*\|_2^2.
\end{aligned} \tag{3.104}$$

It further implies that the sequence $\{\|\mathbf{x}^k - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^k - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^*\|_2^2\}$ is monotonically decreasing and bounded below by 0; hence the sequence must be convergent to a limit, denoted by ξ:

$$\lim_{k \to +\infty} \|\mathbf{x}^k - \mathbf{x}^*\|_2^2 + \|\mathbf{u}^k - \mathbf{u}^*\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\|_2^2 + \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^*\|_2^2 = \xi. \tag{3.105}$$

Taking the limit on both sides of (3.103) yields:

$$\begin{aligned}
&\lim_{k \to +\infty} \|\mathbf{y}^{k+1} - \mathbf{x}^{k+1}\|_2^2 = 0, \qquad &&\lim_{k \to +\infty} \|\mathbf{y}^{k+1} - \mathbf{x}^{k}\|_2^2 = 0, \\
&\lim_{k \to +\infty} \|\mathbf{v}^{k+1} - \mathbf{u}^{k+1}\|_2^2 = 0, &&\lim_{k \to +\infty} \|\mathbf{v}^{k+1} - \mathbf{u}^{k}\|_2^2 = 0, \\
&\lim_{k \to +\infty} \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k+1}\|_2^2 = 0, &&\lim_{k \to +\infty} \|\boldsymbol{\mu}^{k+1} - \boldsymbol{\lambda}^{k}\|_2^2 = 0, \\
&\lim_{k \to +\infty} \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k+1}\|_2^2 = 0, &&\lim_{k \to +\infty} \|\boldsymbol{\nu}^{k+1} - \boldsymbol{\gamma}^{k}\|_2^2 = 0.
\end{aligned} \tag{3.106}$$

Additionally, (3.105) also implies that $\{(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)\}$ is a bounded sequence, which further implies that there exists a sub-sequence $\{(\mathbf{x}^{k_j}, \mathbf{u}^{k_j}, \boldsymbol{\lambda}^{k_j}, \boldsymbol{\gamma}^{k_j})\}$ that converges to a limit point

$(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty)$. We next show that the limit point is indeed a saddle point and is also the unique limit point of $\{(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)\}$. Given any $\mathbf{x} \in \mathbb{X}$ and $\mathbf{u} \in \mathbb{R}^{n_2}$, we have:

$$
\begin{aligned}
&2\rho^{k+1}\Big[\mathcal{L}(\mathbf{x}^{k+1}, \mathbf{u}^{k+1}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1}) - \mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1})\Big] \\
=&2\rho^{k+1}\Big\{\Big[\frac{1}{2}(\mathbf{x}^{k+1})^T P_0 \mathbf{x}^{k+1} - \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x}\Big] + \mathbf{q}_0^T(\mathbf{x}^{k+1} - \mathbf{x}) + \mathbf{c}_0^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+ \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\Big[\frac{1}{2}(\mathbf{x}^{k+1})^T P_{\mathrm{i}} \mathbf{x}^{k+1} - \frac{1}{2}\mathbf{x}^T P_{\mathrm{i}} \mathbf{x}\Big] + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\mathbf{q}_{\mathrm{i}}^T(\mathbf{x}^{k+1} - \mathbf{x}) \\
&+ \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\mathbf{c}_{\mathrm{i}}^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+\boldsymbol{\nu}^{k+1}A(\mathbf{x}^{k+1} - \mathbf{x}) + \boldsymbol{\nu}^{k+1}B(\mathbf{u}^{k+1} - \mathbf{u})\Big\} \\
=& \underbrace{2\rho^{k+1}\Big[-\frac{1}{2}(\mathbf{x}^{k+1} - \mathbf{x})^T P_0(\mathbf{x}^{k+1} - \mathbf{x}) - \sum_{\mathrm{i}=1}^{m_1}\mu_{\mathrm{i}}^{k+1}\frac{1}{2}(\mathbf{x}^{k+1} - \mathbf{x})^T P_{\mathrm{i}}(\mathbf{x}^{k+1} - \mathbf{x})\Big]}_{(\Delta)} \\
&+2\rho^{k+1}\Big[(P_0\mathbf{x}^{k+1} + \mathbf{q}_0)^T(\mathbf{x}^{k+1} - \mathbf{x}) + \mathbf{c}_0^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+ \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}(P_{\mathrm{i}}\mathbf{x}^{k+1} + \mathbf{q}_{\mathrm{i}})^T(\mathbf{x}^{k+1} - \mathbf{x}) + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\mathbf{c}_{\mathrm{i}}^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+\boldsymbol{\nu}^{k+1}A(\mathbf{x}^{k+1} - \mathbf{x}) + \boldsymbol{\nu}^{k+1}B(\mathbf{u}^{k+1} - \mathbf{u})\Big] \\
\leq&2\rho^{k+1}\Big[(P_0\mathbf{x}^{k+1} + \mathbf{q}_0)^T(\mathbf{x}^{k+1} - \mathbf{x}) + \mathbf{c}_0^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+ \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}(P_{\mathrm{i}}\mathbf{x}^{k+1} + \mathbf{q}_{\mathrm{i}})^T(\mathbf{x}^{k+1} - \mathbf{x}) + \sum_{\mathrm{i}=1}^{m_1} \mu_{\mathrm{i}}^{k+1}\mathbf{c}_{\mathrm{i}}^T(\mathbf{u}^{k+1} - \mathbf{u}) \\
&+\boldsymbol{\nu}^{k+1}A(\mathbf{x}^{k+1} - \mathbf{x}) + \boldsymbol{\nu}^{k+1}B(\mathbf{u}^{k+1} - \mathbf{u})\Big].
\end{aligned} \tag{3.107}
$$

The positive semi-definiteness of each P_{i} for all $\mathrm{i} = 0, 1 \ldots m$ guarantees the non-positiveness of (Δ), which makes the last inequality hold. Applying Lemma 2.2.1 on (3.15c) with $\hat{\mathbf{z}} = (\mathbf{x}^{k+1}, \mathbf{u}^{k+1})$, $\bar{\mathbf{z}} = (\mathbf{x}^k, \mathbf{u}^k)$ and $\mathbf{z} = (\mathbf{x}, \mathbf{u})$ yields:

$$
\begin{aligned}
&2\rho^{k+1}\Big\{(P_0\mathbf{y}^{k+1} + \mathbf{q}_0)^T\mathbf{x}^{k+1} + \mathbf{c}_0^T\mathbf{u}^{k+1} + r_0 \\
&\quad + \sum_{\mathrm{i}=1}^{m_1}\mu_{\mathrm{i}}^{k+1}\Big[(P_{\mathrm{i}}\mathbf{y}^{k+1} + \mathbf{q}_{\mathrm{i}})^T\mathbf{x}^{k+1} + \mathbf{c}_{\mathrm{i}}^T\mathbf{u}^{k+1} + r_{\mathrm{i}}\Big] \\
&\quad + \boldsymbol{\nu}^{k+1}(A\mathbf{x}^{k+1} + B\mathbf{u}^{k+1} - \mathbf{b})\Big\} \\
-&2\rho^{k+1}\Big\{(P_0\mathbf{y}^{k+1} + \mathbf{q}_0)^T\mathbf{x} + \mathbf{c}_0^T\mathbf{u} + r_0
\end{aligned}
$$

$$
\begin{aligned}
&\qquad + \sum_{\mathrm{i}=1}^{m} \mu_{\mathrm{i}}^{k+1}\Big[(P_{\mathrm{i}}\mathbf{y}^{k+1} + \mathbf{q}_{\mathrm{i}})^T\mathbf{x} + \mathbf{c}_{\mathrm{i}}^T\mathbf{u} + r_{\mathrm{i}}\Big] \\
&\qquad + \boldsymbol{\nu}^{k+1}(A\mathbf{x} + B\mathbf{u} - \mathbf{b})\Big\} \\
&\le \|\mathbf{x}^k - \mathbf{x}\|_2^2 - \|\mathbf{x}^{k+1} - \mathbf{x}\|_2^2 - \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_2^2 \\
&+ \|\mathbf{u}^k - \mathbf{u}\|_2^2 - \|\mathbf{u}^{k+1} - \mathbf{u}\|_2^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 \\
&\le \Big(\|\mathbf{x}^k - \mathbf{x}^{k+1}\|_2^2 + \|\mathbf{x}^{k+1} - \mathbf{x}\|_2^2\Big) - \|\mathbf{x}^{k+1} - \mathbf{x}\|_2^2 - \|\mathbf{x}^{k+1} - \mathbf{x}^k\|_2^2 \\
&+ \Big(\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_2^2 + \|\mathbf{u}^{k+1} - \mathbf{u}\|_2^2\Big) - \|\mathbf{u}^{k+1} - \mathbf{u}\|_2^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0.
\end{aligned} \tag{3.108}
$$

Adding the above two inequalities yields

$$
\begin{aligned}
&2\rho^{k+1}\Big[\mathcal{L}(\mathbf{x}^{k+1}, \mathbf{u}^{k+1}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1}) - \mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\mu}^{k+1}, \boldsymbol{\nu}^{k+1})\Big] \\
&+2\rho^{k+1}\Big\{(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_0(\mathbf{x}^{k+1} - \mathbf{x}) \\
&\qquad + \sum_{\mathrm{i}=1}^{m} \mu_{\mathrm{i}}^{k+1}\Big[(\mathbf{y}^{k+1} - \mathbf{x}^{k+1})^T P_{\mathrm{i}}(\mathbf{x}^{k+1} - \mathbf{x})\Big]\Big\} \le 0.
\end{aligned} \tag{3.109}
$$

Taking the limits over an appropriate sub-sequence $\{k_{\mathrm{j}}\}$ on both sides and using (3.106), we have:

$$
\mathcal{L}(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty) \le \mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty), \quad \forall \mathbf{x} \in \mathbb{X}, \forall \mathbf{u} \in \mathbb{R}^{n_2}. \tag{3.110}
$$

Similarly, given any $\boldsymbol{\lambda} \in \mathbb{R}_+^{m_1}$ and $\boldsymbol{\gamma} \in \mathbb{R}^{m_2}$, applying Lemma 2.2.1 on (3.15d) with $\widehat{\mathbf{z}} = (\boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})$, $\bar{\mathbf{z}} = (\boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)$ and $\mathbf{z} = (\boldsymbol{\lambda}, \boldsymbol{\gamma})$ yields

$$
\begin{aligned}
&2\rho^{k+1}\Big[\mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}, \boldsymbol{\lambda}, \boldsymbol{\gamma}) - \mathcal{L}(\mathbf{y}^{k+1}, \mathbf{v}^{k+1}, \boldsymbol{\lambda}^{k+1}, \boldsymbol{\gamma}^{k+1})\Big] \\
&\le \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}\|_2^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}\|_2^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 \\
&+ \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 \\
&\le \Big(\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k+1}\|_2^2 - \|\boldsymbol{\lambda}^{k+1} + \boldsymbol{\lambda}\|_2^2\Big) - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}\|_2^2 - \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^k\|_2^2 \\
&+ \Big(\|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^{k+1}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} + \boldsymbol{\gamma}\|_2^2\Big) - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}\|_2^2 - \|\boldsymbol{\gamma}^{k+1} - \boldsymbol{\gamma}^k\|_2^2 = 0.
\end{aligned} \tag{3.111}
$$

Taking the limits over an appropriate sub-sequence $\{k_{\mathrm{j}}\}$ on both sides and using (3.106), we have:

$$
\mathcal{L}(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}, \boldsymbol{\gamma}) \le \mathcal{L}(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty), \quad \forall \boldsymbol{\lambda} \in \mathbb{R}_+^{m_1}, \forall \boldsymbol{\gamma} \in \mathbb{R}^{m_2}. \tag{3.112}
$$

Therefore, we show that $(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty)$ is indeed a saddle point of the Lagrangian function $\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, \boldsymbol{\gamma})$. Then (3.105) implies that

$$\lim_{k\to+\infty} \|\mathbf{x}^k - \mathbf{x}^\infty\|_2^2 + \|\mathbf{u}^k - \mathbf{u}^\infty\|_2^2 + \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^\infty\|_2^2 + \|\boldsymbol{\gamma}^k - \boldsymbol{\gamma}^\infty\|_2^2 = \xi. \tag{3.113}$$

Since we have already argued (after Eq. (3.106)) that there exists a bounded sequence of $\{(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)\}$ that converges to $(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty)$; that is, there exists $\{k_\mathrm{j}\}$ such that $\lim_{k_\mathrm{j}\to+\infty} \|\mathbf{x}^{k_\mathrm{j}} - \mathbf{x}^\infty\|_2^2 + \|\mathbf{u}^{k_\mathrm{j}} - \mathbf{u}^\infty\|_2^2 + \|\boldsymbol{\lambda}^{k_\mathrm{j}} - \boldsymbol{\lambda}^\infty\|_2^2 + \|\boldsymbol{\gamma}^{k_\mathrm{j}} - \boldsymbol{\gamma}^\infty\|_2^2 = 0$, which then implies that $\xi = 0$. Therefore, we show that $\{(\mathbf{x}^k, \mathbf{u}^k, \boldsymbol{\lambda}^k, \boldsymbol{\gamma}^k)\}$ converges globally to a saddle point $(\mathbf{x}^\infty, \mathbf{u}^\infty, \boldsymbol{\lambda}^\infty, \boldsymbol{\gamma}^\infty)$.

$\square$

4. A DISTRIBUTED ALGORITHM FOR MULTI-STAGE STOCHASTIC PROGRAMS WITH APPLICATION TO ELECTRICITY CAPACITY EXPANSION

4.1 Decomposition Methods for Multi-stage Stochastic Program

In previous chapters, we propose distributed algorithms to address the issues of nonlinear coupling constraints, asynchronous iterative scheme and non-separability in both objective function and coupling constraints, when solving large-scale constrained convex optimization problems. In this chapter, we apply the N-block PCPM algorithm to solve electricity capacity expansion models under uncertainty in the form of multi-stage stochastic programs.

4.1.1 Multi-stage Stochastic Program

Multi-stage stochastic programs (MSPs) represent a framework for sequential decision making under uncertainties, where a sequence of uncertain data $(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_T)$ is revealed gradually over stages. The decision variables $\mathbf{x}_t$, corresponding to each stage for $t = 1, \ldots, T$, should be made adaptive to this process. We assume that the sequence $(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_T)$ evolves according to a known stochastic process, and can be regarded as a sequence of random variables with a specified probability distribution. In each stage t, we use $\boldsymbol{\xi}_t$ to denote the random data vector and ξ_t to denote a specific realization; similarly, we use $\boldsymbol{\xi}_{[t]} \coloneqq (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$ to denote the history of the random sequence up to stage t and $\xi_{[t]} \coloneqq (\xi_1, \ldots, \xi_t)$ to denote a specific realization. The decision dynamics is as follows: in each stage t, a realization $\xi_t \in \mathbb{R}^{d_t}$ is observed and followed by a decision variable $\mathbf{x}_t$, whose value, depending on ξ_t and $\mathbf{x}_{t-1}$ from previous stage, is determined to minimize the objective function value of the current stage and the expected value of future stages. A T-stage stochastic program problem can then be expressed into the following nested formulation:

$$\min_{\mathbf{x}_1 \in \mathcal{D}_1} f_1(\mathbf{x}_1) + \mathbb{E}^{\boldsymbol{\xi}_2}\Big[\min_{\mathbf{x}_2 \in \mathcal{X}_2^{\boldsymbol{\xi}_2}(\mathbf{x}_1)} f_2^{\boldsymbol{\xi}_2}(\mathbf{x}_2) + \mathbb{E}^{\boldsymbol{\xi}_3|\boldsymbol{\xi}_{[2]}}\Big[\cdots + \mathbb{E}^{\boldsymbol{\xi}_T|\boldsymbol{\xi}_{[T-1]}}\Big[\min_{\mathbf{x}_T \in \mathcal{X}_T^{\boldsymbol{\xi}_T}(\mathbf{x}_{T-1})} f_T^{\boldsymbol{\xi}_T}(\mathbf{x}_T)\Big]\Big]\Big], \quad (4.1)$$

where $\mathbf{x}_t \in \mathbb{R}^{n_t}$ is the decision variable for each stage $t = 1, \ldots, T$. $f_1 : \mathbb{R}^{n_1} \to \mathbb{R}$ is a continuous function, and $f_t^{\boldsymbol{\xi}_t} : \mathbb{R}^{n_t} \to \mathbb{R}$ is a continuous convex function depending on $\boldsymbol{\xi}_t$

for $t = 2, \ldots, T$. $\mathcal{D}_1 \subset \mathbb{R}^{n_1}$ is a deterministic convex set, and the abstract constraint set $\mathcal{X}_t^{\boldsymbol{\xi}_t}(\mathbf{x}_{t-1})$ for $t = 2, \ldots, T$ can be written out explicitly, for example, as a linear equality constraint:

$$\mathcal{X}_t^{\boldsymbol{\xi}_t}(\mathbf{x}_{t-1}) = \{\mathbf{x}_t \in \mathcal{D}_t | A_t^{\boldsymbol{\xi}_t}\mathbf{x}_{t-1} + B_t^{\boldsymbol{\xi}_t}\mathbf{x}_t = \mathbf{c}_t^{\boldsymbol{\xi}_t}\},$$

where $\mathcal{D}_t \subset \mathbb{R}^{n_t}$ is a deterministic convex set, $A_t^{\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t \times n_{t-1}}$ and $B_t^{\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t \times n_t}$ are matrices depending on $\boldsymbol{\xi}_t$, and $\mathbf{c}_t^{\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t}$ is a vector depending on $\boldsymbol{\xi}_t$. Note that the explicit form of $\mathcal{X}_t^{\boldsymbol{\xi}_t}(\mathbf{x}_{t-1})$ is certainly not limited to linear equality constraints. However, as a starting point, and for the ease of presenting the detailed design and implementation of our algorithm, throughout this chapter, we only consider linearly constrained MSPs. The notation $\mathbb{E}^{\boldsymbol{\xi}_t|\boldsymbol{\xi}_{[t-1]}}$ denotes the conditional expectation operation with respect to $\boldsymbol{\xi}_t$ given $\boldsymbol{\xi}_{[t-1]}$ for $t = 2, \ldots, T$. Note that ξ_1 is observed at the very beginning of the decision process and thus is regarded as a deterministic data, so the conditional expectation $\mathbb{E}^{\boldsymbol{\xi}_2|\boldsymbol{\xi}_{[1]}}[\cdots]$ is simply equivalent to $\mathbb{E}^{\boldsymbol{\xi}_2}[\cdots]$.

4.1.2 Scenario Tree and Node Separability

Computational approach for solving MSPs (4.1) usually is based on a discretization of the underlying stochastic process, or more specifically, an approximation with a finite number of realizations, which can be depicted in the form of a *scenario tree* [41]. Such an approximation may be constructed by various methods [42]–[45].

Let $\mathcal{T}$ denote the set of all *nodes* in a scenario tree associated with a discretized stochastic process $(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_T)$. The tree has a set of a finite number of nodes organized at each *level* corresponding to each stage for all $t = 1, \ldots, T$, denoted as $\mathcal{I}_t$. At level 1, there is only one *root node*, associated with the deterministic data ξ_1. From level 2, there are as many nodes as many different observations of the random data $\boldsymbol{\xi}_t$ that may occur after each realization of $\boldsymbol{\xi}_{t-1}$ from the previous level. The tree is grown recursively in such a way up to level T. Let $t(\mathrm{i})$ denote the level containing each node $\mathrm{i} \in \mathcal{T}$, and the total number of nodes, denoted by N, is finite.

Given a node i and its *child node* i_c, an *edge* connecting them is associated with a probability $p_{\mathrm{i},\mathrm{i}_c}$, denoting the conditional probability of an observation of the random data

$\boldsymbol{\xi}_{t(\mathrm{i}_c)}$ at node i_c, given a realization of $\boldsymbol{\xi}_{[t(\mathrm{i})]}$ at node i. Let p_{i}^{nd} denote the probability of each node for all $\mathrm{i} = 1, \ldots, N$. Given that $p_1^{nd} = 1$, we obtain $p_{\mathrm{i}}^{nd} = p_{a(\mathrm{i})}^{nd} \times p_{a(\mathrm{i}),\mathrm{i}}$ for $\mathrm{i} = 2, \ldots, N$, where $a(\mathrm{i})$ denotes the unique *ancestor node*. Take a 4-stage stochastic process as an example, which is depicted in a scenario tree illustrated in Figure 4.1. There are 16 nodes in the 4-level

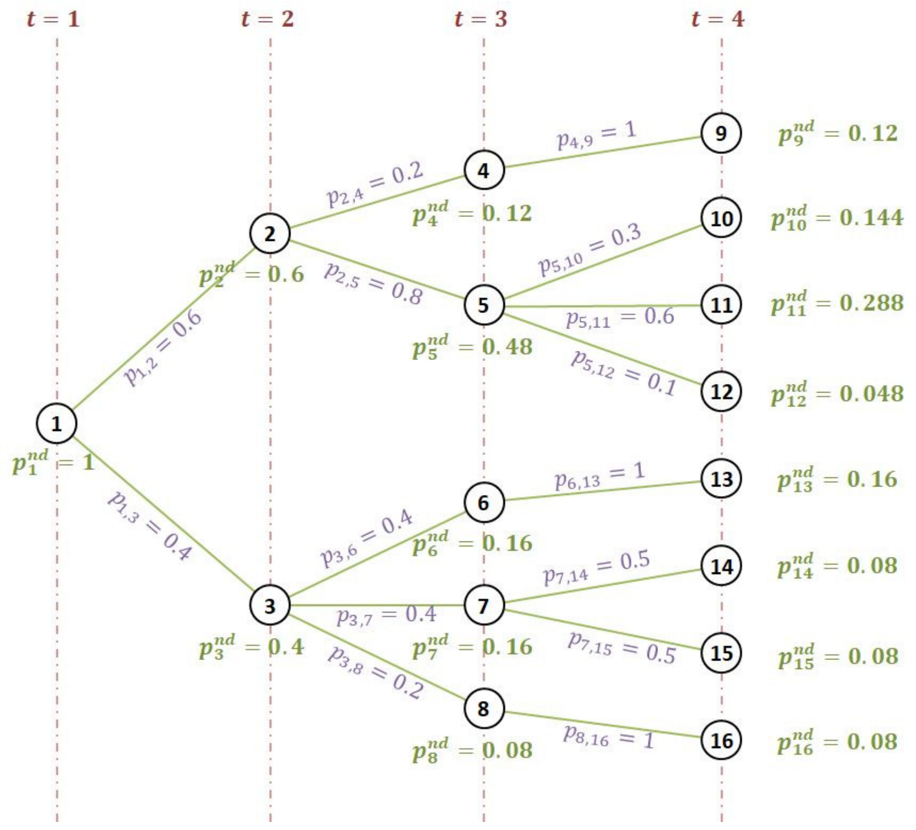


Figure 4.1. An example of a 4-stage stochastic process depicted in the form of a 4-level scenario tree.

tree and the probability of each node is calculated and shown in the figure.

A T-stage stochastic program of form (4.1) can then be reformulated as the following deterministic, large-scale, constrained convex optimization problem with **Node Separability**:

$$\begin{aligned}
\underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & \sum_{\mathrm{i}=1}^{N} p_{\mathrm{i}}^{nd} f_{\mathrm{i}}(\mathbf{x}_{\mathrm{i}}) \\
\text{subject to} \quad & \mathbf{x}_{\mathrm{i}} \in \mathcal{D}_{t(\mathrm{i})}, \quad \mathrm{i} = 1, \ldots, N, \\
& A_{\mathrm{i}}\mathbf{x}_{a(\mathrm{i})} + B_{\mathrm{i}}\mathbf{x}_{\mathrm{i}} = \mathbf{c}_{\mathrm{i}}, \quad \mathrm{i} = 2, \ldots, N,
\end{aligned} \tag{4.2}$$

where $\mathbf{x}_{\mathrm{i}} \in \mathbb{R}^{n_{t(\mathrm{i})}}$ is the decision variable associated with each node $\mathrm{i} = 1, \ldots, N$. $f_{\mathrm{i}} : \mathbb{R}^{n_{t(\mathrm{i})}} \to \mathbb{R}$ is a continuous convex function for all $\mathrm{i} = 1, \ldots, N$. $A_{\mathrm{i}} \in \mathbb{R}^{m_{t(\mathrm{i})} \times n_{t(a(\mathrm{i}))}}$ and $B_{\mathrm{i}} \in \mathbb{R}^{m_{t(\mathrm{i})} \times n_{t(\mathrm{i})}}$ are deterministic matrices for $\mathrm{i} = 2, \ldots, N$, and $\mathbf{c}_{\mathrm{i}} \in \mathbb{R}^{m_{t(\mathrm{i})}}$ is a deterministic vector for $\mathrm{i} = 2, \ldots, N$.

4.1.3 Nonanticipativity and Scenario Separability

At the final level T of a scenario tree, each node $\mathrm{i} \in \mathcal{I}_T$ is called a *leaf node*. A *scenario* is a path from the root node to a leaf node. Let $\mathcal{S}$ denote the set of all scenarios, and the total number of scenarios, denoted by S, is finite. Each scenario $s \in \mathcal{S}$ represents a possible realization of the entire stochastic process $(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_T)$. The probability of each scenario, denoted by p_s^{sc}, is the multiplication of probabilities associated with arcs visited along the path from the root node to each leaf node. For example in Figure 4.1, there are 8 scenarios in total, corresponding to the 8 paths from the root node 1 to leaf nodes $\mathrm{i} = 9, \ldots, 16$. The probability of each scenario is the same as of each leaf node.

Given a finite number of scenarios, we can regard the whole decision sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_T)$ as a mapping of each scenario to the whole decision space $\mathbb{R}^{n_{seq}}$, where $n_{seq} = \sum_{t=1}^{T} n_t$; that is, assigning to each scenario a vector $\mathbf{x}^s := \begin{pmatrix} \mathbf{x}_1^s \\ \vdots \\ \mathbf{x}_T^s \end{pmatrix} \in \mathbb{R}^{n_{seq}}$ for all $s = 1, \ldots, S$, where $\mathbf{x}_t^s \in \mathbb{R}^{n_t}$ for all $t = 1, \ldots, T$. Accordingly, the objective function for each scenario $f^s : \mathbb{R}^{n_{seq}} \to \mathbb{R}$ is defined as $f^s(\mathbf{x}^s) := \sum_{t=1}^{T} f_t^s(\mathbf{x}_t^s)$ for all $s = 1, \ldots, S$, where $f_t^s : \mathbb{R}^{n_t} \to \mathbb{R}$ is a continuous convex function for all $t = 1, \ldots, T$. For each scenario $s = 1, \ldots, S$, the constraint set of the decision sequence $\mathbf{x}^s$ can be denoted by

$$\mathcal{X}^s := \left\{ \mathbf{x}^s \in \prod_{t=1}^{T} \mathcal{D}_t \,\middle|\, A_t^s \mathbf{x}_{t-1}^s + B_t^s \mathbf{x}_t = \mathbf{c}_t^s, \quad t = 2, \ldots, T \right\},$$

where $A_t^s \in \mathbb{R}^{m_t \times n_{t-1}}$ and $B_t^s \in \mathbb{R}^{m_t \times n_t}$ are deterministic matrices for $t = 2, \ldots, T$, and $\mathbf{c}_t^s \in \mathbb{R}^{m_t}$ is a deterministic vector for $t = 2, \ldots, T$.

The decision sequence $\mathbf{x}^s$ for each scenario $s = 1, \ldots, S$ should also satisfy the *nonanticipativity constraints*. At each stage t, the scenario set $\mathcal{S}$ can be partitioned into finitely many disjoint subsets, where each scenario is observationally indistinguishable, based on the realization up to stage t. Such a subset is called a *scenario bundle* and denoted by $\mathcal{V}$. The set containing all scenario bundles at stage t is denoted by $\mathcal{U}_t$ for all $t = 1, \ldots, T$. The nonanticipativity constraints basically require $\mathbf{x}_t^s = \mathbf{x}_t^{s'}$, given any two scenarios $s, s' \in \mathcal{V}$ and any scenario bundle $\mathcal{V} \in \mathcal{U}_t$ for all $t = 1, \ldots, T$. As illustrated in Figure 4.2, there are

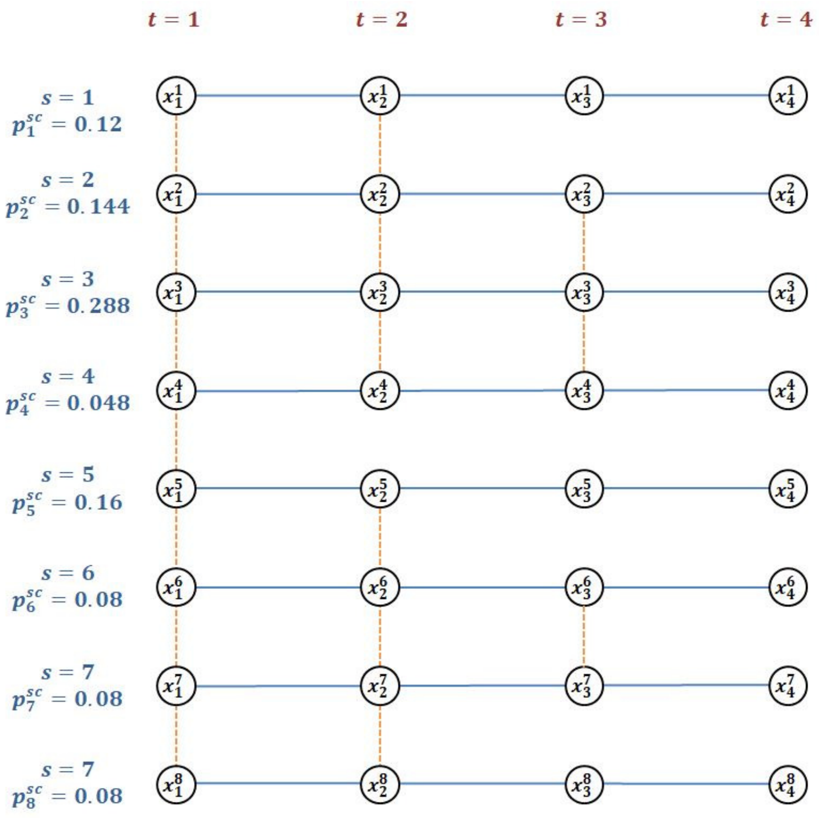


Figure 4.2. The decision sequence for each scenario of the 4-level scenario tree in Figure 4.1. Vertical dotted lines represent the nonanticipativity constraints.

8 decision sequences assigned to the 8 scenarios of the 4-level scenario tree in Figure 4.1. The vertical dotted lines represent the nonaticipativity constraints. For example, at stage $t = 3$, $\mathcal{U}_3 = \Big\{\{1\}, \{2, 3, 4\}, \{5\}, \{6, 7\}, \{8\}\Big\}$; hence we have the nonantipativity constraints: $\mathbf{x}_3^2 = \mathbf{x}_3^3 = \mathbf{x}_3^4$ and $\mathbf{x}_3^6 = \mathbf{x}_3^7$.

Let a vector $\vec{\mathbf{x}} \in \mathbb{R}^{n_{sc}}$ denote decision sequences of all scenarios: $\vec{\mathbf{x}} := \begin{pmatrix} \mathbf{x}^1 \\ \vdots \\ \mathbf{x}^S \end{pmatrix}$, and $n_{sc} = S \times n_{seq}$. Let $\mathcal{N}$ denote a subspace containing the vector $\vec{\mathbf{x}}$, where the nonanticipativity constraints are satisfied:

$$\mathcal{N} := \left\{ \vec{\mathbf{x}} \in \mathbb{R}^{n_{sc}} \middle| \mathbf{x}_t^s = \mathbf{x}_t^{s'}, \quad \forall s, s' \in \mathcal{V}, \quad \forall \mathcal{V} \in \mathcal{U}_t, \quad t = 1, \ldots, T \right\}.$$

Let J denote an operator that maps an arbitrary vector $\vec{\mathbf{x}} \in \mathbb{R}^{n_{sc}}$ to a unique vector $\vec{\mathbf{v}} := J\vec{\mathbf{x}}$, which is of the same dimension as $\vec{\mathbf{x}}$ and can be calculated as:

$$\mathbf{v}_t^s = \frac{\sum_{s \in \mathcal{V}} \mathbf{x}_t^s}{|\mathcal{V}|}, \quad \forall s \in \mathcal{V}, \quad \forall \mathcal{V} \in \mathcal{U}_t, \quad t = 1, \ldots, T,$$

where $|\mathcal{V}|$ denotes the total number of elements in the set $\mathcal{V}$. For example, in Figure 4.2, take $\mathcal{V} = \{2, 3, 4\} \in \mathcal{U}_3$ at stage $t = 3$, and $|\mathcal{V}| = 3$. Clearly, $\vec{\mathbf{v}} \in \mathcal{N}$, and the linear operator J is called an *aggregation operator* that maps from the space $\mathbb{R}^{n_{sc}}$ to the subspace $\mathcal{N}$. The nonanticipativity constraints $\vec{\mathbf{x}} \in \mathcal{N}$ can then be reformulated as $I\vec{\mathbf{x}} = J\vec{\mathbf{x}}$, where I denotes the identity operator. Denoting $K := I - J$, the nonanticipativity constraints can be further rewritten as: $K\vec{\mathbf{x}} = \mathbf{0}$, and the linear operator K can be represented using a matrix $(K_1 \cdots K_S)$ where $K_s \in \mathbb{R}^{n_{sc} \times n_{seq}}$ for all $s = 1, \ldots, S$.

A T-stage stochastic program of form (4.1) can be reformulated as the following deterministic, large-scale, constrained convex optimization problem with **Scenario Separability**:

$$\begin{aligned} \underset{\mathbf{x}^1, \ldots, \mathbf{x}^S}{\text{minimize}} \quad & \sum_{s=1}^{S} p_s^{sc} f^s(\mathbf{x}^s) \\ \text{subject to} \quad & \mathbf{x}^s \in \mathcal{X}^s, \quad s = 1, \ldots, S, \\ & \sum_{s=1}^{S} K_s \mathbf{x}^s = \mathbf{0}. \end{aligned} \tag{4.3}$$

4.1.4 Comparison of Node Decomposition and Scenario Decomposition

In the previous two subsections, we have showed the reformulated form of an MSP with node separability (4.2) and scenario separability (4.3) respectively. In both of the two formulations, the decision variables can be separated into multiple blocks, and the entire large-scale problem can be decomposed into multiple sub-problems to be solved in parallel using a distributed algorithm. For example, PHA [46] is widely used as a scenario decomposition algorithm.

In this subsection, we make a comparison between the possible decomposition methods based on the two different types of separability. As listed in Table 4.1, in the node decomposition method, the large-scale problem can be decomposed all the way down by each decision variable associated with a node; while in the scenario decomposition method, it can only be decomposed down by each decision sequence assigned to a scenario, which has a much larger size that might cause memory storage issues in big-data applications. However, the explicit nonantipativity constraints make the decision sequences decoupled from the stage linking constraints, which in return brings a notable benefit.

Table 4.1. Comparison of Node and Scenario Decomposition

	Node Decomposition	**Scenario Decomposition**
Decompose By	node	scenario
Smallest Decomposed Unit	$\mathbf{x}^{\mathrm{i}}$	$\mathbf{x}^{s}$
Size of Smallest Decomposed Unit	$n_{t(\mathrm{i})}$ (**pro**)	$n_{seq} = \sum_{t=1}^{T} n_t$ (**con**)
Number of Decomposed Unit	$N = \mathcal{O}(\mathrm{e}^T)$	$S = \mathcal{O}(\mathrm{e}^T)$
Coupled with Stage Linking Constraint?	yes (**con**)	no (**pro**)
Type of Nonanticipativity Constraint	implicit	explicit

4.2 A Hybrid Decomposition Method for Multi-scale Multi-stage Stochastic Program under Multi-scale Uncertainties

4.2.1 Additional Structures

In many real-world applications, the decision-making process can be mathematically modeled as an MSP in the formulation of (4.1). Additionally, it may have other structures that can lead to efficient algorithm design, such as the ones listed below.

(i) the decision vector $\mathbf{x}_t$ can be separated into two sub-vectors as $\mathbf{x}_t := \begin{pmatrix} \mathbf{y}_t \\ \mathbf{z}_t \end{pmatrix}$ for all $t = 1, \ldots, T$, where the sub-vector $\mathbf{y}_t \in \mathbb{R}^{n_t^Y}$ is called *aggregate level decision*, the sub-vector $\mathbf{z}_t \in \mathbb{R}^{n_t^Z}$ is called *detailed level decision*, and $n_t = n_t^Y + n_t^Z$,

(ii) the feasible region $\mathcal{D}_t$ can be separated as $\mathcal{D}_t = \mathcal{D}_t^Y \times \mathcal{D}_t^Z$ for all $t = 1, \ldots, T$, where $\mathcal{D}_t^Y \subset \mathbb{R}^{n_t^Y}$ and $\mathcal{D}_t^Z \subset \mathbb{R}^{n_t^Z}$ are convex sets,

(iii) the objective function $f_1(\mathbf{x}_1)$ can be separated as $f_1(\mathbf{x}_1) = f_1^Y(\mathbf{y}_1) + f_1^Z(\mathbf{z}_1)$, where $f_1^Y : \mathbb{R}^{n_1^Y} \to \mathbb{R}$ and $f_1^Z : \mathbb{R}^{n_1^Z} \to \mathbb{R}$ are continuous convex functions, and the objective function $f_t^{\boldsymbol{\xi}_t}(\mathbf{x}_t)$ can be separated as $f_t^{\boldsymbol{\xi}_t}(\mathbf{x}_t^s) = f_t^{Y,\boldsymbol{\xi}_t}(\mathbf{y}_t) + f_t^{Z,\boldsymbol{\xi}_t}(\mathbf{z}_t)$ for $t = 2, \ldots, T$, where $f_t^{Y,\boldsymbol{\xi}_t} : \mathbb{R}^{n_t^Y} \to \mathbb{R}$ and $f_t^{Z,\boldsymbol{\xi}_t} : \mathbb{R}^{n_t^Z} \to \mathbb{R}$ are continuous convex functions depending on $\boldsymbol{\xi}_t$,

(iv) the linear equality constraint $A_t^{\boldsymbol{\xi}_t}\mathbf{x}_{t-1} + B_t^{\boldsymbol{\xi}_t}\mathbf{x}_t = \mathbf{c}_t^{\boldsymbol{\xi}_t}$ can be rewritten as

$$\begin{pmatrix} A_t^{Y,\boldsymbol{\xi}_t} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{z}_{t-1} \end{pmatrix} + \begin{pmatrix} B_t^{Y,\boldsymbol{\xi}_t} & 0 \\ B_t^{LY,\boldsymbol{\xi}_t} & B_t^{LZ,\boldsymbol{\xi}_t} \end{pmatrix} \begin{pmatrix} \mathbf{y}_t \\ \mathbf{z}_t \end{pmatrix} = \begin{pmatrix} \mathbf{c}_t^{Y,\boldsymbol{\xi}_t} \\ \mathbf{c}_t^{L,\boldsymbol{\xi}_t} \end{pmatrix}$$

for $t = 2, \ldots, T$, where $A_t^{Y,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^Y \times n_{t-1}^Y}$, $B_t^{Y,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^Y \times n_t^Y}$, $B_t^{LY,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^L \times n_t^Y}$ and $B_t^{LZ,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^L \times n_t^Z}$ are matrices depending on $\boldsymbol{\xi}_t$, and $\mathbf{c}_t^{Y,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^Y}$ and $\mathbf{c}_t^{L,\boldsymbol{\xi}_t} \in \mathbb{R}^{m_t^L}$ are vectors depending on $\boldsymbol{\xi}_t$.

In such a structure, it is observed that only the aggregated level decision $\mathbf{y}_t$'s are coupled in the stage linking constraint: $A_t^{Y,\boldsymbol{\xi}_t}\mathbf{y}_{t-1} + B_t^{Y,\boldsymbol{\xi}_t}\mathbf{y}_t = \mathbf{c}_t^{Y,\boldsymbol{\xi}_t}$; while each detailed level decision $\mathbf{z}_t$ is stagewise independent but coupled with $\mathbf{y}_t$ at the same stage through a linking constraint $B_t^{LY,\boldsymbol{\xi}_t}\mathbf{y}_t + B_t^{LZ,\boldsymbol{\xi}_t}\mathbf{z}_t = \mathbf{c}_t^{L,\boldsymbol{\xi}_t}$.

4.2.2 Extended Nonanticipativity and Hybrid Scenario-node Decomposition

Using the similar ways we show in Section 4.1, such an MSP can certainly be reformulated into a deterministic, large-scale, constrained convex optimization problem with either node or scenario separability, which has its own strength and shortcoming listed in Table 4.1. To reap the benefits of both methods, while overcoming their drawbacks, we propose a novel reformulation where the **y**-part is with scenario separability and the **z**-part is with node separability.

To each scenario of a scenario tree, we assign a decision sequence $\mathbf{y}^s := \begin{pmatrix} \mathbf{y}_1^s \\ \vdots \\ \mathbf{y}_T^s \end{pmatrix} \in \mathbb{R}^{n_{seq}^Y}$ for all $s = 1, \dots, S$, and $n_{seq}^Y = \sum_{t=1}^T n_t^Y$. Accordingly, an objective function for each scenario $f^{Y,s} : \mathbb{R}^{n_{seq}^Y} \to \mathbb{R}$ is defined as $f^{Y,s}(\mathbf{y}^s) := \sum_{t=1}^T f_t^{Y,s}(\mathbf{y}_t^s)$ for all $s = 1, \dots, S$, where $f_t^{Y,s} : \mathbb{R}^{n_t^Y} \to \mathbb{R}$ is a continuous convex function for all $t = 1, \dots, T$. For each scenario $s = 1, \dots, S$, the constraint set of the decision sequence $\mathbf{y}^s$ is denoted by

$$\mathcal{Y}^s := \left\{ \mathbf{y}^s \in \prod_{t=1}^T \mathcal{D}_t^Y \middle| A_t^{Y,s} \mathbf{y}_{t-1}^s + B_t^{Y,s} \mathbf{y}_t = \mathbf{c}_t^{y,s}, \quad t = 2, \dots, T \right\},$$

where $A_t^{Y,s} \in \mathbb{R}^{m_t \times n_{t-1}}$ and $B_t^{Y,s} \in \mathbb{R}^{m_t \times n_t}$ are deterministic matrices for $t = 2, \dots, T$, and $\mathbf{c}_t^{Y,s} \in \mathbb{R}^{m_t}$ is a deterministic vector for $t = 2, \dots, T$.

With each node $\mathrm{i} = 1, \dots, N$ in the scenario tree, a decision variable $\mathbf{z}_\mathrm{i} \in \mathbb{R}^{n_{t(\mathrm{i})}^Z}$ and a local copy $\mathbf{y}_\mathrm{i}^{nd} \in \mathbb{R}^{n_{t(\mathrm{i})}^Y}$ are associated, as well as a continuous convex objective function $f_\mathrm{i}^Z : \mathbb{R}^{n_{t(\mathrm{i})}^Z} \to \mathbb{R}$. The decision variables $\mathbf{z}_\mathrm{i}$ and $\mathbf{y}_\mathrm{i}^{nd}$ are denoted together by a vector $\mathbf{w}_\mathrm{i}^{nd} := \begin{pmatrix} \mathbf{y}_\mathrm{i}^{nd} \\ \mathbf{z}_\mathrm{i} \end{pmatrix} \in \mathbb{R}^{n_{t(\mathrm{i})}}$. The original linkage between the aggregated level decision $\mathbf{y}_t$ and the detailed level decision $\mathbf{z}_t$ can then be absorbed in the single constraint set:

$$\mathcal{W}_\mathrm{i}^{nd} := \{ \mathbf{y}_\mathrm{i}^{nd} \in \mathbb{R}^{n_{t(\mathrm{i})}^Y}, \mathbf{z}_\mathrm{i} \in \mathcal{D}_{t(\mathrm{i})}^Z | B_\mathrm{i}^{LY} \mathbf{y}_\mathrm{i}^{nd} + B_\mathrm{i}^{LZ} \mathbf{z}_\mathrm{i} = \mathbf{c}_\mathrm{i}^L \}$$

for all $\mathrm{i} = 1, \dots, N$, where $B_\mathrm{i}^{LY} \in \mathbb{R}^{m_{t(\mathrm{i})}^L \times n_{t(\mathrm{i})}^Y}$ and $B_\mathrm{i}^{LZ} \in \mathbb{R}^{m_{t(\mathrm{i})}^L \times n_{t(\mathrm{i})}^Z}$ are deterministic matrices, and $\mathbf{c}_\mathrm{i}^L \in \mathbb{R}^{m_{t(\mathrm{i})}^L}$ is a deterministic vector. As a trade-off, the local copy $\mathbf{y}_\mathrm{i}^{nd}$ associated with each node is additionally required to be equal to the value of $\mathbf{y}_t^s$'s in a corresponding scenario bundle.

Let us consider the 4-level scenario tree in Figure 4.1 again. As illustrated in Figure 4.3, at each stage t, each scenario bundle $\mathcal{V} \in \mathcal{U}_t$ is now vertically extended to a corresponding

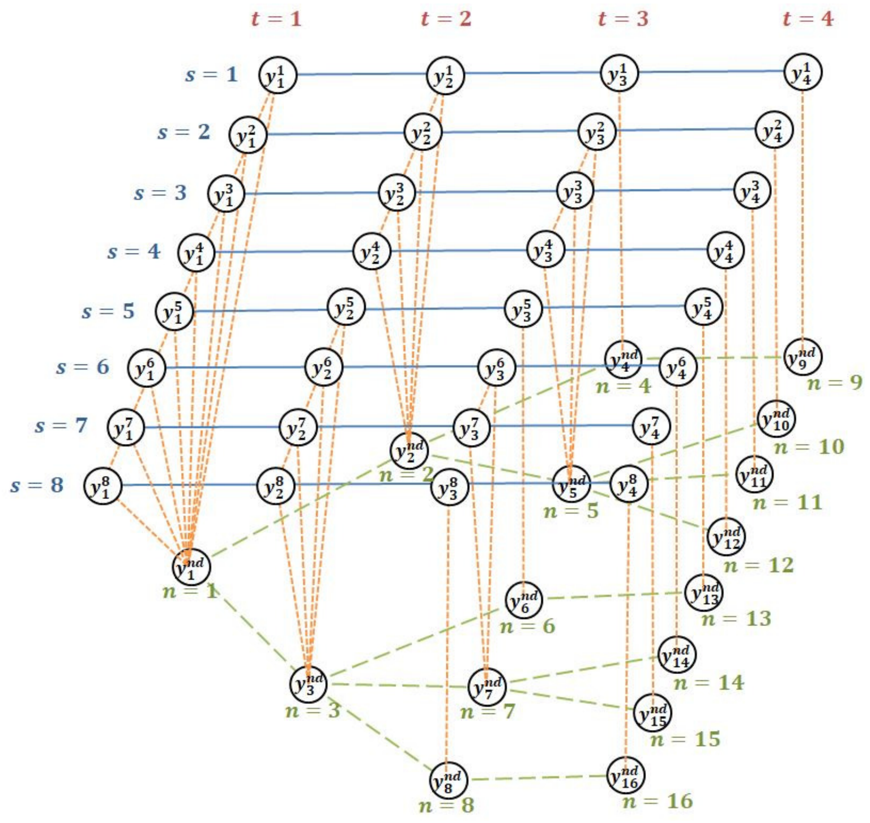


Figure 4.3. The decision sequence for each scenario and the local copy at each node of the 4-level scenario tree in Figure 4.1. Vertical dotted lines represent the extended nonanticipativity constraints.

node i at the same level. For example, the scenario bundle $\mathcal{V} = \{6, 7\} \in \mathcal{U}_3$ at stage 3 is extended to a hybrid bundle $\mathcal{V}^{sn} = \{s = 6, s = 7, n = 7\}$. Instead of simply requiring $\mathbf{y}_3^6 = \mathbf{y}_3^7$, we now require $\mathbf{y}_3^6 = \mathbf{y}_3^7 = \mathbf{y}_7^{nd}$. We call such a hybrid bundle a *scenario-node bundle*, denoted by $\mathcal{V}^{sn}$, and the set containing all scenario-node bundles at stage t is denoted by $\mathcal{U}_t^{sn}$ for all $t = 1, \ldots, T$.

Let a vector $\vec{\mathbf{y}}^{sn} \in \mathbb{R}^{n_{sc}^Y + n_{nd}^Y}$ denote all $\mathbf{y}$-variables: $\vec{\mathbf{y}}^{sn} := \begin{pmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^S \\ \mathbf{y}_1^{nd} \\ \vdots \\ \mathbf{y}_N^{nd} \end{pmatrix}$, and $n_{sc}^Y = S \times n_{seq}^Y$ and $n_{nd}^Y = \sum_{\mathrm{i}=1}^N n_{t(\mathrm{i})}^Y$. Let $\mathcal{N}^{sn}$ denote a subspace containing the vector $\vec{\mathbf{y}}^{sn}$, where the extended nonanticipativity constraints are satisfied; namely,

$$\mathcal{N}^{sn} := \left\{ \vec{\mathbf{y}}^{sn} \in \mathbb{R}^{n_{sc}^Y + n_{nd}^Y} \middle| \begin{array}{ll} \mathbf{y}_t^s = \mathbf{y}_t^{s'}, & \forall s, s' \in \mathcal{V}^{sn}, \\ \mathbf{y}_t^s = \mathbf{y}_{\mathrm{i}}^{nd}, & \forall s, \mathrm{i} \in \mathcal{V}^{sn}, \end{array} \quad \forall \mathcal{V}^{sn} \in \mathcal{U}_t^{sn}, \quad t = 1, \ldots, T \right\}.$$

Let J^{sn} denote the aggregation operator that maps from the space $\mathbb{R}^{n_{sc}^Y + n_{nd}^Y}$ to the subspace $\mathcal{N}^{sn}$, and the mapped vector $\vec{\mathbf{v}}^{sn} := J^{sn} \vec{\mathbf{y}}^{sn}$, of the same dimension as $\vec{\mathbf{y}}^{sn}$, can be calculated as:

$$\begin{array}{ll} \mathbf{v}_t^s = \frac{\sum_{s \in \mathcal{V}^{sn}} \mathbf{y}_t^s + \sum_{\mathrm{i} \in \mathcal{V}^{sn}} \mathbf{y}_{\mathrm{i}}^{nd}}{|\mathcal{V}|}, & \forall s \in \mathcal{V}^{sn}, \\ \mathbf{v}_{\mathrm{i}}^{nd} = \frac{\sum_{s \in \mathcal{V}^{sn}} \mathbf{y}_t^s + \sum_{\mathrm{i} \in \mathcal{V}^{sn}} \mathbf{y}_{\mathrm{i}}^{nd}}{|\mathcal{V}|}, & \forall \mathrm{i} \in \mathcal{V}^{sn}, \end{array} \quad \forall \mathcal{V}^{sn} \in \mathcal{U}_t^{sn}, \quad t = 1, \ldots, T.$$

In a similar way as in Section 4.1.3, the extended nonanticipativity constraints $\vec{\mathbf{y}}^{sn} \in \mathcal{N}^{sn}$ can then be reformulated as $I \vec{\mathbf{y}}^{sn} = J^{sn} \vec{\mathbf{y}}^{sn}$. Denoting $K^{sn} := I - J^{sn}$, the extended nonanticipativity constraints can be further rewritten as: $K^{sn} \vec{\mathbf{y}}^{sn} = \mathbf{0}$, and the linear operator K^{sn} can be represented using a matrix $(K_1, \ldots, K_S, K_1^{nd}, \ldots, K_N^{nd})$ where each $K_s \in \mathbb{R}^{(n_{sc}^Y + n_{nd}^Y) \times n_{seq}^Y}$ for all $s = 1, \ldots, S$, and each $K_{\mathrm{i}}^{nd} \in \mathbb{R}^{(n_{sc}^Y + n_{nd}^Y) \times n_{t(\mathrm{i})}^Y}$ for all $\mathrm{i} = 1, \ldots, N$.

A T-stage stochastic program of form (4.1) with the additional structure stated in Section 4.2.1 can then be reformulated as the following deterministic, large-scale, constrained convex optimization problem with **Hybrid Scenario-node Separability**:

$$\begin{array}{ll} \underset{\substack{\mathbf{y}^1, \ldots, \mathbf{y}^S \\ \mathbf{w}_1^{nd}, \ldots, \mathbf{w}_N^{nd}}}{\text{minimize}} & \sum_{s=1}^S p_s^{sc} f^{Y,s}(\mathbf{y}^s) + \sum_{\mathrm{i}=1}^N p_{\mathrm{i}}^{nd} f_{\mathrm{i}}^Z(\mathbf{z}_{\mathrm{i}}) \\ \text{subject to} & \mathbf{y}^s \in \mathcal{Y}^s, \quad s = 1, \ldots, S, \\ & \mathbf{w}_{\mathrm{i}}^{nd} \in \mathcal{W}_{\mathrm{i}}^{nd}, \quad \mathrm{i} = 1, \ldots, N, \\ & \sum_{s=1}^S K_s \mathbf{y}^s + \sum_{\mathrm{i}=1}^N K_{\mathrm{i}}^{nd} \mathbf{y}_{\mathrm{i}}^{nd} = \mathbf{0}. \end{array} \tag{4.4}$$

Compared with the node-only and scenario-only decomposition methods, the hybrid scenario-node decomposition inherits both of their strengths but none of their shortcomings, as listed in Table 4.2. The entire problem is decomposed all the way down by the smallest

Table 4.2. Comparison of Hybrid Scenario-node Decomposition with Node-only and Scenario-only Decomposition

	Node Decomposition	**Scenario Decomposition**	**Hybrid Scenario-node Decomposition**
Decompose By	node only	scenario only	scenario and node
Smallest Decomposed Unit	$\begin{pmatrix} \mathbf{y} \\ \mathbf{z} \end{pmatrix}^{\mathrm{i}}$	$\begin{pmatrix} \mathbf{y} \\ \mathbf{z} \end{pmatrix}^{s}$	$\mathbf{y}^s$ and $\mathbf{w}^{\mathrm{i}}$
Size of Smallest Decomposed Unit	$n_{t(\mathrm{i})}$ (**pro**)	$n_{seq} = \sum_{t=1}^{T} n_t$ (**con**)	$n_{seq}^{Y} = \sum_{t=1}^{T} n_t^{Y}$ and $n_{t(\mathrm{i})}$ (**pro**)
Number of Decomposed Unit	$N = \mathcal{O}(\mathrm{e}^T)$	$S = \mathcal{O}(\mathrm{e}^T)$	$S + N = \mathcal{O}(\mathrm{e}^T)$
Coupled with Stage Linking Constraint?	yes (**con**)	no (**pro**)	no (**pro**)
Type of Nonanticipativity Constraint	implicit	explicit	explicit and extended

separate blocks of the decision variables, which are $\mathbf{y}^s$ for all $s = 1, \ldots, S$ and $\mathbf{w}_{\mathrm{i}}^{nd}$ for all $\mathrm{i} = 1, \ldots, N$. Note that the size of the aggregated level decision variable $\mathbf{y}_t$ is usually much smaller than that of the detailed level decision variable $\mathbf{z}_t$; i.e., $n_t^Y \ll n_t^Z$. For example, in an electricity capacity expansion model, $\mathbf{y}_t$ represents the yearly planning variables while $\mathbf{z}_t$ represents the hourly operational variables, whose size can be 10^3 times of $\mathbf{y}_t$. In this case, $n_{seq}^Y = \sum_{t=1}^{T} n_t^Y \ll n_{t(\mathrm{i})} = n_{t(\mathrm{i})}^Y + n_{t(\mathrm{i})}^Z$. Each decomposed unit, either $\mathbf{y}^s$ or $\mathbf{w}_{\mathrm{i}}^{nd}$ is not coupled with any stage linking constraint, making the resulting deterministic problem suitable for massive parallel computing.

4.2.3 Under Multi-scale Uncertainties

In many applications, not only the size of the detailed level decision variable is huge, but also various uncertainties of different temporal scales are involved. At each node of a scenario tree, assume that the decision variable $\mathbf{z}_{\mathrm{i}}$ additionally depends on an uncertainty $\boldsymbol{\eta}_{\mathrm{i}}$

of different temporal scale from $\boldsymbol{\xi}_t$ for all $\mathrm{i} = 1, \ldots, N$, which is further assumed to have a finite number of realizations $r = 1, \ldots, R_\mathrm{i}$ with probability p_r^{rl}.

Similarly, with each realization of $\boldsymbol{\eta}_\mathrm{i}$ at node i, a decision variable $\mathbf{z}_{\mathrm{i},r} \in \mathbb{R}^{n^Z_{t(\mathrm{i})}}$ and a local copy $\mathbf{y}^{nr}_{\mathrm{i},r} \in \mathbb{R}^{n^Y_{t(\mathrm{i})}}$ are associated, as well as a continuous convex objective function $f^Z_{\mathrm{i},r} : \mathbb{R}^{n^Z_{t(\mathrm{i})}} \to \mathbb{R}$ for all $r = 1, \ldots, R_\mathrm{i}$ and $\mathrm{i} = 1, \ldots, N$. The decision variables $\mathbf{z}_{\mathrm{i},r}$ and $\mathbf{y}^{nr}_{\mathrm{i},r}$ are denoted together by a vector $\mathbf{w}^{nr}_{\mathrm{i},r} := \begin{pmatrix} \mathbf{y}^{nr}_{\mathrm{i},r} \\ \mathbf{z}_{\mathrm{i},r} \end{pmatrix} \in \mathbb{R}^{n_{t(\mathrm{i})}}$, subject to a constraint set:

$$\mathcal{W}^{nr}_{\mathrm{i},r} := \{\mathbf{y}^{nr}_{\mathrm{i},r} \in \mathbb{R}^{n^Y_{t(\mathrm{i})}}, \mathbf{z}_{\mathrm{i},r} \in \mathcal{D}^Z_{t(\mathrm{i})} | B^{LY}_{\mathrm{i},r} \mathbf{y}^{nr}_{\mathrm{i},r} + B^{LZ}_{\mathrm{i},r} \mathbf{z}_{\mathrm{i},r} = \mathbf{c}^L_{\mathrm{i},r}\}$$

for all $r = 1, \ldots, R_\mathrm{i}$ and $\mathrm{i} = 1, \ldots, N$, where $B^{LY}_{\mathrm{i},r} \in \mathbb{R}^{m^L_{t(\mathrm{i})} \times n^Y_{t(\mathrm{i})}}$ and $B^{LZ}_{\mathrm{i},r} \in \mathbb{R}^{m^L_{t(\mathrm{i})} \times n^Z_{t(\mathrm{i})}}$ are deterministic matrices, and $\mathbf{c}^L_{\mathrm{i},r} \in \mathbb{R}^{m^L_{t(\mathrm{i})}}$ is a deterministic vector.

Still consider the 4-level scenario tree in Figure 4.1, and assume that $\boldsymbol{\eta}_\mathrm{i}$ at each node i has 4 possible realizations. As illustrated in Figure 4.4, the scenario bundle $\mathcal{V} = \{6, 7\} \in \mathcal{U}_3$ at stage 3 is further extended to a hybrid *scenario-node-realization bundle* $\mathcal{V}^{snr} = \{s = 6, s = 7, (\mathrm{i}, r) = (7, 1), (\mathrm{i}, r) = (7, 2), (\mathrm{i}, r) = (7, 3), (\mathrm{i}, r) = (7, 4)\}$. The extended nonanticipativity constraints become: $\mathbf{y}^6_3 = \mathbf{y}^7_3 = \mathbf{y}^{nr}_{7,1} = \mathbf{y}^{nr}_{7,2} = \mathbf{y}^{nr}_{7,3} = \mathbf{y}^{nr}_{7,4}$. The set containing all scenario-node-realization bundles at stage t is denoted by $\mathcal{U}^{snr}_t$ for all $t = 1, \ldots, T$.

Similarly, let a vector $\vec{\mathbf{y}}^{snr} \in \mathbb{R}^{n^Y_{sc} + n^Y_{nr}}$ denote all $\mathbf{y}$-variables: $\vec{\mathbf{y}}^{snr} := \begin{pmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^S \\ \mathbf{y}^{nr}_{1,1} \\ \vdots \\ \mathbf{y}^{nr}_{N,R_N} \end{pmatrix}$, and $n^Y_{sc} = S \times n^Y_{seq}$ and $n^Y_{nr} = \sum_{\mathrm{i}=1}^N \left[R_\mathrm{i} n^Y_{t(\mathrm{i})} \right]$. Let $\mathcal{N}^{snr}$ denote a subspace containing the vector $\vec{\mathbf{y}}^{snr}$ where the extended nonanticipativity constraints are satisfied as follows:

$$\begin{aligned} \mathcal{N}^{snr} := \Bigg\{ \vec{\mathbf{y}}^{snr} \in \mathbb{R}^{n^Y_{sc} + n^Y_{nr}} \Bigg| \begin{array}{ll} \mathbf{y}^s_t = \mathbf{y}^{s'}_t, & \forall s, s' \in \mathcal{V}^{snr}, \\ \mathbf{y}^{nr}_{\mathrm{i},r} = \mathbf{y}^{nr}_{\mathrm{i},r'}, & \forall (\mathrm{i}, r), (\mathrm{i}, r') \in \mathcal{V}^{snr}, \\ \mathbf{y}^s_t = \mathbf{y}^{nr}_{\mathrm{i},r}, & \forall s, (\mathrm{i}, r) \in \mathcal{V}^{snr}, \end{array} \\ \forall \mathcal{V}^{snr} \in \mathcal{U}^{snr}_t, \quad t = 1, \ldots, T \Bigg\}. \end{aligned} \tag{4.5}$$

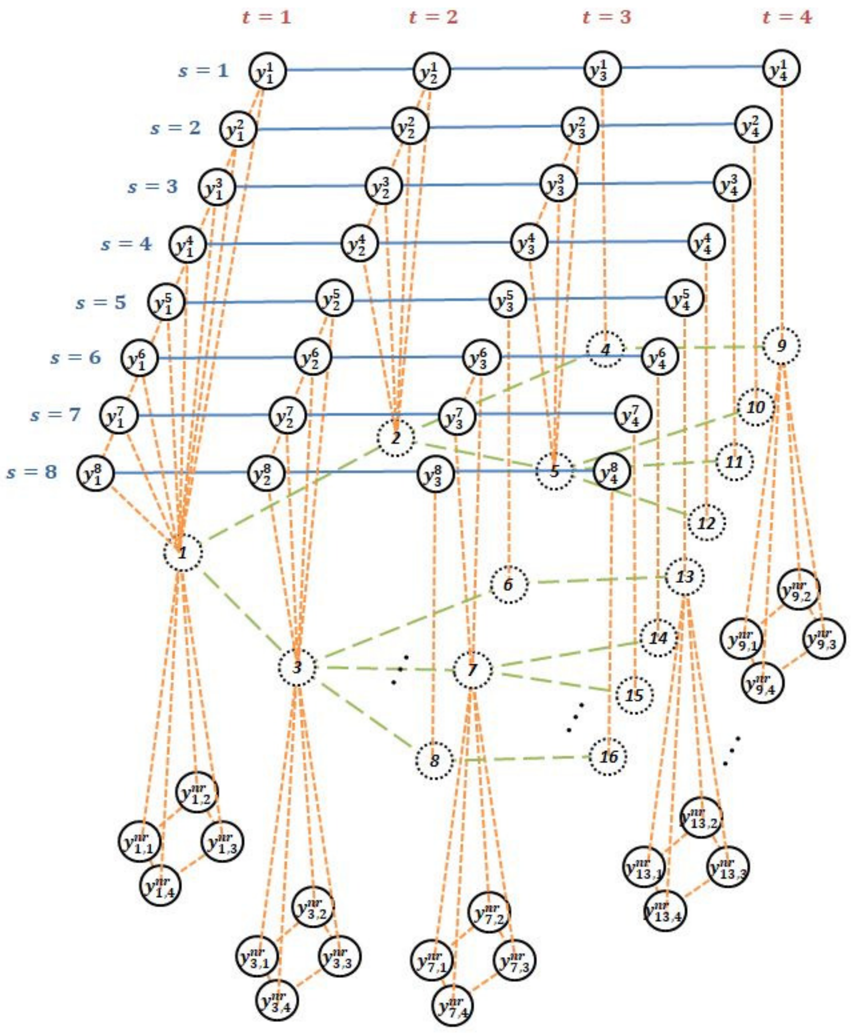


Figure 4.4. The decision sequence for each scenario and the local copy for each realization of the multi-scale uncertainty at each node of the 4-level scenario tree in Figure 4.1. Vertical dotted lines represent the extended nonanticipativity constraints.

Let J^{snr} denote the aggregation operator that maps from the space $\mathbb{R}^{Sn^Y_{seq}+n^Y_{nr}}$ to the subspace $\mathcal{N}^{snr}$, and the mapped vector $\vec{\mathbf{v}}^{snr} := J^{snr}\vec{\mathbf{y}}^{snr}$, of the same dimension as $\vec{\mathbf{y}}^{snr}$, can be calculated as:

$$\left.\begin{array}{ll} \mathbf{v}_t^s = \frac{\sum_{s\in\mathcal{V}^{snr}} \mathbf{y}_t^s + \sum_{(\mathrm{i},r)\in\mathcal{V}^{snr}} \mathbf{y}_{\mathrm{i},r}^{nr}}{|\mathcal{V}|}, & \forall s \in \mathcal{V}^{snr}, \\ \mathbf{v}_{\mathrm{i},r}^{nr} = \frac{\sum_{s\in\mathcal{V}^{snr}} \mathbf{y}_t^s + \sum_{(\mathrm{i},r)\in\mathcal{V}^{snr}} \mathbf{y}_{\mathrm{i},r}^{nr}}{|\mathcal{V}|}, & \forall (\mathrm{i},r) \in \mathcal{V}^{snr}, \end{array}\right. \quad \forall \mathcal{V}^{snr} \in \mathcal{U}_t^{snr}, \quad t = 1, \ldots, T.$$

In a similar way as in Section 4.1.3, the extended nonanticipativity constraints $\vec{\mathbf{y}}^{snr} \in \mathcal{N}^{snr}$ can then be reformulated as $I\vec{\mathbf{y}}^{snr} = J^{snr}\vec{\mathbf{y}}^{snr}$. Denoting $K^{snr} := I - J^{snr}$, the extended nonanticipativity constraints can be further rewritten as: $K^{snr}\vec{\mathbf{y}}^{snr} = \mathbf{0}$, and the linear operator K^{snr} can be represented using a matrix $(K_1, \dots, K_S, K^{nr}_{1,1}, \dots, K^{nr}_{N,R_N})$ where each $K_s \in \mathbb{R}^{(n^Y_{sc}+n^Y_{nr})\times n^Y_{seq}}$ for all $s = 1, \dots, S$, and each $K^{nr}_{\mathrm{i},r} \in \mathbb{R}^{(n^Y_{sc}+n^Y_{nd})\times n^Y_{t(\mathrm{i})}}$ for all $r = 1, \dots, R_\mathrm{i}$ and $\mathrm{i} = 1, \dots, N$.

Under multi-scale uncertainties, a T-stage stochastic program of form (4.1) with the additional structure stated in Section 4.2.1 can be reformulated as the following deterministic, large-scale, constrained convex optimization problem with **Hybrid Scenario-node-realization Separability**:

$$\begin{aligned} \underset{\substack{\mathbf{y}^1,\dots,\mathbf{y}^S \\ \mathbf{w}^{nr}_{1,1},\dots,\mathbf{w}^{nr}_{N,R_N}}}{\text{minimize}} \quad & \sum_{s=1}^{S} p^{sc}_s f^{Y,s}(\mathbf{y}^s) + \sum_{\mathrm{i}=1}^{N} p^{nd}_\mathrm{i} \Big[\sum_{r=1}^{R_\mathrm{i}} p^{rl}_r f^Z_{\mathrm{i},r}(\mathbf{z}_{\mathrm{i},r}) \Big] \\ \text{subject to} \quad & \mathbf{y}^s \in \mathcal{Y}^s, \quad s = 1, \dots, S, \\ & \mathbf{w}^{nr}_{\mathrm{i},r} \in \mathcal{W}^{nr}_{\mathrm{i},r}, \quad r = 1, \dots, R_\mathrm{i}, \quad \mathrm{i} = 1, \dots, N, \\ & \sum_{s=1}^{S} K_s \mathbf{y}^s + \sum_{\mathrm{i}=1}^{N} \sum_{r=1}^{R_\mathrm{i}} K^{nr}_{\mathrm{i},r} \mathbf{y}^{nr}_{\mathrm{i},r} = \mathbf{0}. \end{aligned} \tag{4.6}$$

4.3 A Simplified PCPM Algorithm using Orthogonal Projection

The decision variables of the optimization problem (4.6) can be divided into two groups: $\vec{\mathbf{y}} := \begin{pmatrix} \mathbf{y}^1 \\ \vdots \\ \mathbf{y}^S \end{pmatrix}$ and $\vec{\mathbf{w}} := \begin{pmatrix} \mathbf{w}^{nr}_{1,1} \\ \vdots \\ \mathbf{w}^{nr}_{N,R_N} \end{pmatrix}$. The vector $\vec{\mathbf{y}}$ can be further separated into blocks of $\mathbf{y}^s$ for all $s = 1, \dots, S$, and the vector $\vec{\mathbf{w}}$ can be further separated into blocks of $\mathbf{w}^{nr}_{\mathrm{i},r} = \begin{pmatrix} \mathbf{y}^{nr}_{\mathrm{i},r} \\ \mathbf{z}_{\mathrm{i},r} \end{pmatrix}$ for all $r = 1, \dots, R_\mathrm{i}$ and $\mathrm{i} = 1, \dots, N$. The objective function in problem (4.6) is the summation of a set of separable functions defined on all blocks, coupled by the extended nonanticipativity constraints, which are linear equality constraints. We can apply the ADMM algorithm or the PCPM algorithm to solve problem (4.6) in a distributed fashion.

Let $\vec{\boldsymbol{\lambda}} \in \mathbb{R}^{n^Y_{sc}+n^Y_{nr}}$ denote the Lagrangian multiplier, associated with the extended nonanticipativity constraint, in the form of $\vec{\boldsymbol{\lambda}} = \begin{pmatrix} \boldsymbol{\lambda}^1 \\ \vdots \\ \boldsymbol{\lambda}^S \\ \boldsymbol{\lambda}^{nr}_{1,1} \\ \vdots \\ \boldsymbol{\lambda}^{nr}_{N,R_N} \end{pmatrix}$. The classic Lagrangian function $\mathcal{L} : \left(\prod_{s=1}^S \mathcal{Y}^s \right) \times \left(\prod_{\mathrm{i}=1}^N \prod_{r=1}^{R_\mathrm{i}} \mathcal{W}^{nr}_{\mathrm{i},r} \right) \times \mathbb{R}^{n^Y_{sc}+n^Y_{nr}} \to \mathbb{R}$ is defined as:

$$\begin{aligned} \mathcal{L}(\vec{\mathbf{y}}, \vec{\mathbf{w}}, \vec{\boldsymbol{\lambda}}) = & \sum_{s=1}^S p^{sc}_s f^{Y,s}(\mathbf{y}^s) + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} \left[p^{nd}_\mathrm{i} p^{rl}_r f^Z_{\mathrm{i},r}(\mathbf{z}_{\mathrm{i},r}) \right] \\ & + \vec{\boldsymbol{\lambda}}^T \left(\sum_{s=1}^S K_s \mathbf{y}^s + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} K^{nr}_{\mathrm{i},r} \mathbf{y}^{nr}_{\mathrm{i},r} \right), \\ & \forall \mathbf{y}^s \in \mathcal{Y}^s, \quad s = 1, \dots, S, \\ & \forall \mathbf{w}^{nr}_{\mathrm{i},r} \in \mathcal{W}^{nr}_{\mathrm{i},r}, \quad r = 1, \dots, R_\mathrm{i}, \quad \mathrm{i} = 1, \dots, N. \end{aligned} \tag{4.7}$$

It is well-known that for a convex problem of the specific form in (4.6), finding an optimal solution is equivalent to finding a saddle point $(\vec{\mathbf{y}}^*, \vec{\mathbf{w}}^*, \vec{\boldsymbol{\lambda}}^*)$ such that $\mathcal{L}(\vec{\mathbf{y}}^*, \vec{\mathbf{w}}^*, \vec{\boldsymbol{\lambda}}) \leq \mathcal{L}(\vec{\mathbf{y}}^*, \vec{\mathbf{w}}^*, \vec{\boldsymbol{\lambda}}^*) \leq \mathcal{L}(\vec{\mathbf{y}}, \vec{\mathbf{w}}, \vec{\boldsymbol{\lambda}}^*)$. We assume that such a saddle point always exists for problem (4.6).

4.3.1 Apply the PCPM Algorithm

For the ease of argument, we write out again the deterministic equivalent formula from an MSP below, which is the same as (4.6).

$$\begin{aligned} \underset{\substack{\mathbf{y}^s \in \mathcal{Y}^s \\ \mathbf{w}^{nr}_{\mathrm{i},r} \in \mathcal{W}^{nr}_{\mathrm{i},r}}}{\text{minimize}} \quad & \sum_{s=1}^S p^{sc}_s f^{Y,s}(\mathbf{y}^s) + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} \left[p^{nd}_\mathrm{i} p^{rl}_r f^Z_{\mathrm{i},r}(\mathbf{z}_{\mathrm{i},r}) \right] \\ \text{subject to} \quad & \sum_{s=1}^S K_s \mathbf{y}^s + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} K^{nr}_{\mathrm{i},r} \mathbf{y}^{nr}_{\mathrm{i},r} = \mathbf{0}. \qquad (\vec{\boldsymbol{\lambda}}) \end{aligned} \tag{4.8}$$

To apply the PCPM algorithm, at each iteration k, with a given primal-dual pair, $(\vec{\mathbf{y}}^k, \vec{\mathbf{w}}^k, \vec{\boldsymbol{\lambda}}^k)$, we start with a dual predictor update:

$$\vec{\mathbf{p}}^{k+1} := \vec{\boldsymbol{\lambda}}^k + \rho\Big(\sum_{s=1}^{S} K_s \mathbf{y}^{s,k} + \sum_{\mathrm{i}=1}^{N}\sum_{r=1}^{R_\mathrm{i}} K_{\mathrm{i},r}^{nr}\mathbf{y}_{\mathrm{i},r}^{nr,k}\Big). \tag{4.9}$$

After the dual predictor update, we update the primal variables $(\vec{\mathbf{y}}^{k+1}, \vec{\mathbf{w}}^{k+1})$ by minimizing the Lagrangian function $\mathcal{L}(\vec{\mathbf{y}}, \vec{\mathbf{w}}, \vec{\mathbf{p}}^{k+1})$ evaluated at the dual predictor variable $\vec{\mathbf{p}}^{k+1}$, plus the proximal terms. The primal minimization step can be written as

$$\mathbf{y}^{s,k+1} = \underset{\mathbf{y}^s \in \mathcal{Y}^s}{\operatorname{argmin}}\ p_s^{sc} f^{Y,s}(\mathbf{y}^s) + (\vec{\mathbf{p}}^{k+1})^T K_s \mathbf{y}^s + \frac{1}{2\rho}\|\mathbf{y}^s - \mathbf{y}^{s,k}\|_2^2, \quad s = 1, \ldots, S, \tag{4.10a}$$

$$\begin{aligned}\mathbf{w}_{\mathrm{i},r}^{nr,k+1} = \underset{\mathbf{w}_{\mathrm{i},r}^{nr} \in \mathcal{W}_{\mathrm{i},r}^{nr}}{\operatorname{argmin}}\ & p_\mathrm{i}^{nd} p_r^{rl} f_{\mathrm{i},r}^{Z}(\mathbf{z}_{\mathrm{i},r}) + (\vec{\mathbf{p}}^{k+1})^T K_{\mathrm{i},r}^{nr}\mathbf{y}_{\mathrm{i},r}^{nr} + \frac{1}{2\rho}\|\mathbf{y}_{\mathrm{i},r}^{nr} - \mathbf{y}_{\mathrm{i},r}^{nr,k}\|_2^2 \\ & + \frac{1}{2\rho}\|\mathbf{z}_{\mathrm{i},r} - \mathbf{z}_{\mathrm{i},r}^{k}\|_2^2, \quad r = 1, \ldots, R_\mathrm{i}, \quad \mathrm{i} = 1, \ldots, N.\end{aligned} \tag{4.10b}$$

A dual corrector update is then performed for each Lagrangian multiplier:

$$\vec{\boldsymbol{\lambda}}^{k+1} = \vec{\boldsymbol{\lambda}}^k + \rho\Big(\sum_{s=1}^{S} K_s \mathbf{y}^{s,k+1} + \sum_{\mathrm{i}=1}^{N}\sum_{r=1}^{R_\mathrm{i}} K_{\mathrm{i},r}^{nr}\mathbf{y}_{\mathrm{i},r}^{nr,k+1}\Big). \tag{4.11}$$

4.3.2 Orthogonal Projection

The convergence of the PCPM algorithm to an optimal solution under proper assumptions is well analyzed in [7]. However, we do find an implementation scheme that simplifies the primal minimization step, based on a nice property of the aggregation operator J^{snr}. We first write out the well-known definition of an *orthogonal projection* in our context.

Definition 4.3.1 (Orthogonal Projection)**.** Given a vector space $\mathbb{V}$ equipped with an inner product and a subspace $\mathbb{W}$, consider an operator $P : \mathbb{V} \to \mathbb{W}$ that maps a vector $\mathbf{v} \in \mathbb{V}$ to a unique vector $\mathbf{u} \in \mathbb{W}$. P is called a *projection* if $P = P^2$. P is further called an *orthogonal projection* if $\mathbf{v} - P\mathbf{v} \perp \mathbb{W}$ for any $\mathbf{v} \in \mathbb{V}$, i.e., $\langle \mathbf{v} - P\mathbf{v}, \mathbf{w}\rangle = 0$ for any $\mathbf{v} \in \mathbb{V}$ and $\mathbf{w} \in \mathbb{W}$.

Then, we present the following lemma, showing a nice property of the orthogonal projection.

Lemma 4.3.1. Given a vector space $\mathbb{V}$ equipped with an inner product and a subspace $\mathbb{W}$, if $P : \mathbb{V} \to \mathbb{W}$ is an orthogonal projection, then we have

$$\langle P\mathbf{v}, \mathbf{v}' \rangle = \langle P\mathbf{v}, P\mathbf{v}' \rangle = \langle \mathbf{v}, P\mathbf{v}' \rangle, \quad \forall \mathbf{v}, \mathbf{v}' \in \mathbb{V}.$$

Proof. The first equality holds because $\langle P\mathbf{v}, \mathbf{v}' \rangle = \langle P\mathbf{v}, (\mathbf{v}' - P\mathbf{v}') + P\mathbf{v}' \rangle = \langle P\mathbf{v}, \mathbf{v}' - P\mathbf{v}' \rangle + \langle P\mathbf{v}, P\mathbf{v}' \rangle = \langle P\mathbf{v}, P\mathbf{v}' \rangle$. Similarly, the second equality also holds. $\square$

It is easy to check that the aggregation operator J^{snr} is an orthogonal projection from the space $\mathbb{R}^{n^Y_{sc}+n^Y_{nr}}$ to the subspace $\mathcal{N}^{snr}$, where the extended nonanticipativity constraints are satisfied. Also, $K^{snr} = I - J^{snr}$ is an orthogonal projection from the space $\mathbb{R}^{n^Y_{sc}+n^Y_{nr}}$ to the subspace $\mathcal{N}^{snr\perp}$, where $\mathcal{N}^{snr\perp}$ denotes the subspace orthogonal to $\mathcal{N}^{snr}$.

4.3.3 A Simplified PCPM Algorithm

We observe that the term $\vec{\mathbf{p}}^T\Big(\sum_{s=1}^S K_s \mathbf{y}^s + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} K^{nr}_{\mathrm{i},r}\mathbf{y}^{nr}_{\mathrm{i},r}\Big)$ in primal minimization steps (4.10a) and (4.10b) can be written as an inner product: $\langle \vec{\mathbf{p}}, K^{snr}\vec{\mathbf{y}}^{snr}\rangle$. By Lemma 4.3.1, $\langle \vec{\mathbf{p}}, K^{snr}\vec{\mathbf{y}}^{snr}\rangle = \langle K^{snr}\vec{\mathbf{p}}, \vec{\mathbf{y}}^{snr}\rangle$, since K^{snr} is an orthogonal projection. Denoting $\vec{\mathbf{q}} := K^{snr}\vec{\mathbf{p}}$, the inner product $\langle \vec{\mathbf{p}}, K^{snr}\vec{\mathbf{y}}^{snr}\rangle$ can be further rewritten as $\langle \vec{\mathbf{p}}, K^{snr}\vec{\mathbf{y}}^{snr}\rangle = \langle K^{snr}\vec{\mathbf{p}}, \vec{\mathbf{y}}^{snr}\rangle = \langle \vec{\mathbf{q}}, \vec{\mathbf{y}}^{snr}\rangle = \sum_{s=1}^S (\mathbf{q}^s)^T\mathbf{y}^s + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} (\mathbf{q}^{nr}_{\mathrm{i},r})^T \mathbf{y}^{nr}_{\mathrm{i},r}$.

The dual predictor update (4.9) can be written in the form of

$$\vec{\mathbf{p}}^{k+1} = \vec{\boldsymbol{\lambda}}^k + \rho K^{snr}\vec{\mathbf{y}}^{snr}.$$

Applying the orthogonal projection on both sides, we obtain

$$K^{snr}\vec{\mathbf{p}}^{k+1} = K^{snr}\vec{\boldsymbol{\lambda}}^k + \rho K^{snr}K^{snr}\vec{\mathbf{y}}^{snr},$$

which is equivalent to

$$\begin{aligned}\vec{\mathbf{q}}^{k+1} =& \vec{\boldsymbol{\gamma}}^k + \rho K^{snr}\vec{\mathbf{y}}^{snr} \\ =& \vec{\boldsymbol{\gamma}}^k + \rho\Big(\sum_{s=1}^S K_s \mathbf{y}^{s,k} + \sum_{\mathrm{i}=1}^N \sum_{r=1}^{R_\mathrm{i}} K^{nr}_{\mathrm{i},r}\mathbf{y}^{nr,k}_{\mathrm{i},r}\Big),\end{aligned} \tag{4.12}$$

where $\vec{\boldsymbol{\gamma}} := K^{snr}\vec{\boldsymbol{\lambda}} \in \mathcal{N}^{snr\perp}$ is the new Lagrangian multiplier. The primal minimization step (4.10a) and (4.10b) can be simplified as

$$\mathbf{y}^{s,k+1} = \underset{\mathbf{y}^s \in \mathcal{Y}^s}{\operatorname{argmin}} \; p_s^{sc} f^{Y,s}(\mathbf{y}^s) + (\mathbf{q}^{s,k+1})^T \mathbf{y}^s + \frac{1}{2\rho}\|\mathbf{y}^s - \mathbf{y}^{s,k}\|_2^2, \quad s = 1, \dots, S, \tag{4.13a}$$

$$\begin{aligned}\mathbf{w}_{\mathrm{i},r}^{nr,k+1} = \underset{\mathbf{w}_{\mathrm{i},r}^{nr} \in \mathcal{W}_{\mathrm{i},r}^{nr}}{\operatorname{argmin}} \; & p_{\mathrm{i}}^{nd} p_r^{rl} f_{\mathrm{i},r}^{Z}(\mathbf{z}_{\mathrm{i},r}) + (\mathbf{q}_{\mathrm{i},r}^{nr,k+1})^T \mathbf{y}_{\mathrm{i},r}^{nr} + \frac{1}{2\rho}\|\mathbf{y}_{\mathrm{i},r}^{nr} - \mathbf{y}_{\mathrm{i},r}^{nr,k}\|_2^2 \\ & + \frac{1}{2\rho}\|\mathbf{z}_{\mathrm{i},r} - \mathbf{z}_{\mathrm{i},r}^{k}\|_2^2, \quad r = 1, \dots, R_{\mathrm{i}}, \quad \mathrm{i} = 1, \dots, N.\end{aligned} \tag{4.13b}$$

It can be easily observed that all the K-matrices in (4.10a) and (4.10b) no longer show up in (4.13a) and (4.13b), which greatly simplifies the primal minimization steps. Originally, the calculation of the term $\vec{\mathbf{p}}^T K_s$ or $\vec{\mathbf{p}}^T K_{\mathrm{i},r}^{nr}$ needs values of all components of the Lagrangian multiplier $\vec{\mathbf{p}}$, which are stored distributively, and hence requires extra communication among all computing units. Now, for the update of each $\mathbf{y}$-block or $\mathbf{w}$-block, only the value of the corresponding component $\mathbf{q}^s$ or $\mathbf{q}_{\mathrm{i},r}^{nr}$ is required and can be stored locally in the computing unit responsible for $\mathbf{y}^s$ or $\mathbf{w}_{\mathrm{i},r}^{nr}$. There is no need of any communication among all computing units for the primal minimization step (4.13a) and (4.13b). Similarly to the dual predictor update, applying the orthogonal projection on both sides of the dual corrector update (4.11) yields

$$\vec{\boldsymbol{\gamma}}^{k+1} = \vec{\boldsymbol{\gamma}}^{k} + \rho\Big(\sum_{s=1}^{S} K_s \mathbf{y}^{s,k+1} + \sum_{\mathrm{i}=1}^{N}\sum_{r=1}^{R_{\mathrm{i}}} K_{\mathrm{i},r}^{nr}\mathbf{y}_{\mathrm{i},r}^{nr,k+1}\Big). \tag{4.14}$$

The overall simplified structure of the PCPM algorithm is presented in Algorithm 5 below.

4.4 Electricity Capacity Expansion under Multi-scale Uncertainties

We begin this section by presenting a co-optimization model where the long-term electricity capacity expansion is co-optimized with short-term generation and transmission constraints.

Algorithm 5 Simplified PCPM for solving (4.6)

```
1: Initialization choose an arbitrary starting point $(\vec{\mathbf{y}}^0, \vec{\mathbf{w}}^0, \vec{\gamma}^0)$.
2: $k \leftarrow 0$.
3: while termination conditions are not met do
4:     (Dual Predictor Update)
        update $\vec{\mathbf{q}}^{k+1}$ according to (4.12);
5:     (Primal Update)
        update $\vec{\mathbf{y}}^{k+1}$ and $\vec{\mathbf{w}}^{k+1}$ according to (4.13a) and (4.13b);
6:     (Dual Corrector Update)
        update $\vec{\gamma}^{k+1}$ according to (4.14);
7:     $k \leftarrow k + 1$
8: return $(\vec{\mathbf{y}}^k, \vec{\mathbf{w}}^k, \vec{\gamma}^k)$.
```

4.4.1 Capacity Expansion Planning

At the beginning of each year $t = 1, \dots, T$, a decision of an expanded capacity $x_{g,t}$ for each power generator $g \in \mathbb{G}$ has to be made adaptive to a stochastic process, and a cumulative capacity $k_{g,t}$ for each power generator $g \in \mathbb{G}$ is aggregated as $k_{g,t} = k_{g,t-1} + x_{g,t}$ for $t = 2, \dots, T$. We regard each year t as a stage in an MSP. Given a scenario tree with a set of S scenarios, we describe the model of capacity expansion planning. All indices, sets and functions are listed in Table 4.3; the decision variables are listed in Table 4.4. The overnight

Table 4.3. Indices, Sets and Functions for Capacity Expansion Planning

$\mathcal{S}$	set of a finite number of scenarios, indexed s;
p^{sc}_s	probability of each scenario s;
$\mathbb{J}$	set of substations, indexed j;
$\mathbb{G}$	set of power generators, indexed g;
$\hat{\mathbb{J}}$	set of reserve margin regions, indexed $\hat{\text{j}}$;
$\mathcal{J}(\text{j})$	function that maps a substation $\text{j} \in \mathbb{J}$ to a reserve margin region $\hat{\text{j}} \in \hat{\mathbb{J}}$;
T	number of years in the planning horizon, indexed t;

Table 4.4. Decision Variables for Capacity Expansion Planning

$x^s_{g,t}$	expanded capacity of power generator g in year t for scenario s; [MW]
$k^s_{g,t}$	cumulative capacity of power generator g in year t for scenario s; [MW]

investment cost of an expanded capacity is calculated using a quadratic form: $\frac{1}{2}IC^s_{g,t}(x^s_{g,t})^2$

and is levelized through N_g installments in the future for any $g \in \mathbb{G}$ and $t = 1, \ldots, T$, where $IC^s_{g,t}$ is the quadratic coefficient for scenario $s = 1, \ldots, S$. All parameters for capacity expansion planning are listed in Table 4.5. There's also a fixed operation and maintenance

Table 4.5. Parameters for Capacity Expansion Planning

δ	discount factor; $0 < \delta < 1$;
$IC^s_{g,t}$	quadratic coefficient for the overnight investment cost of power generator g in year t for scenario s; [k\$/MW^2]
N_g	number of years for power generator g to pay the overnight investment cost;
FOM_g	fixed O&M cost for power generator g; [k\$/MW]
KG_g	existing capacity of power generator g; [MW]
DF_g	derating factor of power generator g; [%]
$RM_{\hat{\mathrm{j}}}$	reserve margin requirement for region $\hat{\mathrm{j}}$; [%]
$PK^s_{\hat{\mathrm{j}},t}$	peak level of hourly load in reserve margin region $\hat{\mathrm{j}}$ during year t for scenario s. [MWh]

(O&M) cost of the cumulative capacity: $FOM_g k^s_{g,t}$ for any $g \in \mathbb{G}$ and $t = 1, \ldots, T$. Then, the total cost of capacity expansion planning is:

$$TC^{CE} = \sum_{s=1}^{S} p^{sc}_s \left\{ \sum_{g \in \mathbb{G}} \sum_{t=1}^{T} \delta^t \left[\left(\sum_{t': t-N_g \leq t' \leq t} \frac{1}{N_g} \frac{1}{2} IC^s_{g,t'} x^s_{g,t'} \right) + FOM_g k^s_{g,t} \right] \right\}. \tag{4.15}$$

The decision variables have to satisfy the yearly cumulative constraints:

$$\begin{aligned} &k^s_{g,1} = KG_g, \\ &k^s_{g,t} = k^s_{g,t-1} + x^s_{g,t}, \quad t = 2, \ldots, T, \end{aligned} \qquad \forall g \in \mathbb{G}, \quad s = 1, \ldots, S, \tag{4.16}$$

a reserve margin requirement constraint:

$$\begin{aligned} &\sum_{\mathrm{j} \in \mathbb{J}: \mathcal{J}(\mathrm{j}) = \hat{\mathrm{j}}} \ \sum_{g \in \mathbb{G}: \mathcal{G}(g) = \mathrm{j}} DF_g k^s_{g,t} \geq (1 + RM_{\hat{\mathrm{j}}}) PK^s_{\hat{\mathrm{j}},t}, \\ &\forall \hat{\mathrm{j}} \in \hat{\mathbb{J}}, \quad t = 1, \ldots, T, \quad s = 1, \ldots, S, \end{aligned} \tag{4.17}$$

and the non-negativeness constraint:

$$x^s_{g,t}, k^s_{g,t} \geq 0, \quad \forall g \in \mathbb{G}, \quad t = 1, \ldots, T, \quad s = 1, \ldots, S. \tag{4.18}$$

Due to the presence of linkage between t and $t+1$ in (4.16), the expanded capacity $x_{g,t}$ and the cumulative capacity $k_{g,t}$ are regarded as aggregated level decisions, as discussed in Section 4.2.1.

4.4.2 Sub-hourly Economic Dispatch of Generation and Transmission

We divided the set $\mathbb{G}$ into two groups, where $\mathbb{G}^{SR}$ denotes the set of power generators that cannot change their generation output levels within an hour, while $\mathbb{G}^{FR}$ denotes the set of power generators that can change their output levels quickly within an hour (the so-called fast ramping units).

In each year $t = 1, \dots, T$, the generation level $p_{g,h,t}$ of each slow-response generator $g \in \mathbb{G}^{SR}$ has to be decided for each hour $h = 1, \dots, H$, and the generation level $p_{g,m,t}$ of each fast-response generator $g \in \mathbb{G}^{FR}$ has to be decided for for each sub-hour $m = 1, \dots, M$. Consider a set of N nodes of the same scenario tree given in Section 4.4.1, and for each node $\mathrm{i} = 1, \dots, N$, consider a set of R_i realizations of uncertainty $\boldsymbol{\eta}_\mathrm{i}$. In this subsection, we describe the model of sub-hourly economic dispatch of generation and transmission. Table 4.6 summarizes all the indices, sets, and functions for the hourly/sub-hourly economic dispatch model; while the corresponding decision variables are listed in Table 4.7. The generation cost for slow-response generators is: $\sum_{g\in\mathbb{G}^{SR}} \sum_{h=1}^{H} VOM_g p_{g,h,\mathrm{i},r}$ for all $r = 1, \dots, R_\mathrm{i}$ and $\mathrm{i} = 1, \dots, N$, and the generation cost for fast response generators is: $\sum_{g\in\mathbb{G}^{FR}} \sum_{m=1}^{M} \frac{1}{M} VOM_g p_{g,m,\mathrm{i},r}$ for all $r = 1, \dots, R_\mathrm{i}$ and $\mathrm{i} = 1, \dots, N$. All parameters for sub-hourly economic dispatch of generation and transmission are listed in Table 4.8. There is also a penalty cost for the electricity outage if the generation supply could not meet the demand: $\sum_{\mathrm{j}\in\mathbb{J}} \sum_{m=1}^{M} PENo_{\mathrm{j},m,\mathrm{i},r}$ for all $r = 1, \dots, R_\mathrm{i}$ and $\mathrm{i} = 1, \dots, N$. Then, the total cost of sub-hourly economic dispatch of generation and transmission is:

$$
\begin{aligned}
TC^{ED} = \sum_{\mathrm{i}=1}^{N} p_\mathrm{i}^{nd} \Bigg\{ \sum_{r=1}^{R_\mathrm{i}} p_r^{rl} \delta^t \Big[\sum_{g\in\mathbb{G}^{SR}} \sum_{h=1}^{H} VOM_g p_{g,h,\mathrm{i},r} + \sum_{g\in\mathbb{G}^{FR}} \sum_{m=1}^{M} \frac{1}{M} VOM_g p_{g,m,\mathrm{i},r} \\
+ \sum_{\mathrm{j}\in\mathbb{J}} \sum_{m=1}^{M} PENo_{\mathrm{j},m,\mathrm{i},r} \Big] \Bigg\}.
\end{aligned}
\tag{4.19}
$$

Table 4.6. Indices, Sets and Functions for Sub-hourly Economic Dispatch of Generation and Transmission

$\mathcal{T}$	set of nodes in a scenario tree, indexed i;
p^{nd}_{i}	probability of each node i;
R_{i}	number of realizations of uncertainty $\boldsymbol{\eta}_{\mathrm{i}}$ at node i, indexed r;
p^{rl}_{r}	probability of each realization r;
$\mathbb{G}^{SR}$	set of slow-response generators;
$\mathbb{G}^{FR}$	set of fast-response generators;
$\mathcal{G}(g)$	function that maps a power generator g to a substation $\mathrm{j} \in \mathbb{J}$;
$\mathbb{W}$	set of wind resources, indexed w;
$\mathcal{W}(w)$	function that maps a wind resource w to a substation $\mathrm{j} \in \mathbb{J}$;
$\mathbb{D}$	set of demand nodes, indexed d;
$\mathcal{D}(d)$	function that maps a demand node d to a substation $\mathrm{j} \in \mathbb{J}$;
$\mathbb{L}$	set of transmission lines connecting two substations, indexed l;
$\mathcal{L}_o(l)$	function that maps a transmission line l to an origin substation $\mathrm{j}_o \in \mathbb{J}$;
$\mathcal{L}_d(l)$	function that maps a transmission line l to a destination substation $\mathrm{j}_d \in \mathbb{J}$;
H	number of hours in a year, indexed h;
M	number of sub-hours in a year, indexed m;
$\mathcal{M}(m)$	function that maps a sub-hour m to an hour h.

Table 4.7. Decision Variables for Sub-hourly Economic Dispatch of Generation and Transmission

$k^{nr}_{g,\mathrm{i},r}$	local copy of the cumulative capacity $k_{g,t(\mathrm{i})}$ of power generator g for realization r at node i;
$p_{g,h,\mathrm{i},r}$	generation level of slow-response generator $g \in \mathbb{G}^{SR}$ during hour h and year $t(\mathrm{i})$ for realization r at node i;[MW]
$p_{g,m,\mathrm{i},r}$	generation level of fast-response generator $g \in \mathbb{G}^{FR}$ during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;[MW]
$f^{+}_{l,m,\mathrm{i},r}$	transmission flow on line l from substation j_o to j_d during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;[MW]
$f^{-}_{l,m,\mathrm{i},r}$	transmission flow on line l from substation j_d to j_o during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;[MW]
$\theta_{\mathrm{j},m,\mathrm{i},r}$	phase angle at substation j during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;[rad]
$o_{\mathrm{j},m,\mathrm{i},r}$	outage at at substation j during sub-hour m and year $t(\mathrm{i})$ for realization r at node i.[MW]

Table 4.8. Parameters for Sub-hourly Economic Dispatch of Generation and Transmission

VOM_g	variable O&M cost for power generator g; [k\$/MWh]
$a_{w,m,\mathrm{i},r}$	availability factor of wind resource $w \in \mathbb{W}$ during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;
RU_g	ramp-up rate of cumulative capacity of power generator g;[%]
RD_g	ramp-down rate of cumulative capacity of power generator g;[%]
KW_w	capacity of wind resource w; [MW]
$D_{d,m,\mathrm{i},r}$	level of demand node d during sub-hour m and year $t(\mathrm{i})$ for realization r at node i;[MWh]
KL_l	capacity of transmission line l; [MW]
B_l	percentage of energy loss of transmission line l; [%]
Δ_l	susceptance of transmission line l, which equals the reciprocal of the reactance;
PEN	penalty cost of electricity outage. [k\$/MW]

The generation level of each power generator can not exceed its cumulative capacity:

$$\begin{aligned} &0 \le p_{g,h,\mathrm{i},r} \le k^{nr}_{g,\mathrm{i},r}, \quad \forall g \in \mathbb{G}^{SR}, \quad h = 1,\dots,H, \\ &0 \le p_{g,m,\mathrm{i},r} \le k^{nr}_{g,\mathrm{i},r}, \quad \forall g \in \mathbb{G}^{FR}, \quad m = 1,\dots,M, \end{aligned} \quad r = 1,\dots,R_\mathrm{i}, \quad \mathrm{i} = 1,\dots,N, \tag{4.20}$$

while still satisfies the ramping constraints:

$$\begin{aligned} &p_{g,h,\mathrm{i},r} - p_{g,(h-1),\mathrm{i},r} \le RU_g k^{nr}_{g,\mathrm{i},r}, \\ &p_{g,(h-1),\mathrm{i},r} - p_{g,h,\mathrm{i},r} \le RD_g k^{nr}_{g,\mathrm{i},r}, \\ &\forall g \in \mathbb{G}^{SR}, \quad h = 2,\dots,H, \quad r = 1,\dots,R_\mathrm{i}, \quad \mathrm{i} = 1,\dots,N, \end{aligned} \tag{4.21}$$

$$\begin{aligned} &p_{g,m,\mathrm{i},r} - p_{g,(m-1),\mathrm{i},r} \le \frac{1}{M} RU_g k^{nr}_{g,\mathrm{i},r}, \\ &p_{g,(m-1),\mathrm{i},r} - p_{g,m,\mathrm{i},r} \le \frac{1}{M} RD_g k^{nr}_{g,\mathrm{i},r}, \\ &\forall g \in \mathbb{G}^{FR}, \quad m = 2,\dots,M, \quad r = 1,\dots,R_\mathrm{i}, \quad \mathrm{i} = 1,\dots,N. \end{aligned} \tag{4.22}$$

The Kirchhoff's current law (KCL) has to be satisfied:

$$\sum_{d \in \mathbb{D}:\mathcal{D}(d)=\mathrm{j}} D_{d,m,\mathrm{i},r} + \sum_{l \in \mathbb{L}:\mathcal{L}_o(l)=\mathrm{j}} \frac{1}{M} f^{+}_{l,m,\mathrm{i},r} + \sum_{l \in \mathbb{L}:\mathcal{L}_d(l)=\mathrm{j}} \frac{1}{M} f^{-}_{l,m,\mathrm{i},r}$$

$$
\begin{aligned}
&= \sum_{g\in\mathbb{G}^{SR}:\mathcal{G}(g)=\mathrm{j}} \frac{1}{M} p_{g,\mathcal{H}(m),\mathrm{i},r} + \sum_{g\in\mathbb{G}^{FR}:\mathcal{G}(g)=\mathrm{j}} \frac{1}{M} p_{g,m,\mathrm{i},r} + \sum_{w\in\mathbb{W}:\mathcal{W}(w)=\mathrm{j}} \frac{1}{M} a_{w,m,\mathrm{i},r} KW_w \\
&+ \sum_{l\in\mathbb{L}:\mathcal{L}_d(l)=\mathrm{j}} \frac{1}{M}(1-B_l) f^+_{l,m,\mathrm{i},r} + \sum_{l\in\mathbb{L}:\mathcal{L}_o(l)=\mathrm{j}} \frac{1}{M}(1-B_l) f^-_{l,m,\mathrm{i},r} + o_{\mathrm{j},m,\mathrm{i},r}, \\
&\quad \forall \mathrm{j}\in\mathbb{J}, \quad r=1,\dots,R_\mathrm{i}, \quad \mathrm{i}=1,\dots,N,
\end{aligned} \tag{4.23}
$$

as well as the Kirchhoff's Voltage law (KVL):

$$
\begin{aligned}
& f^+_{l,m,\mathrm{i},r} - f^-_{l,m,\mathrm{i},r} = \Delta_l(\theta_{\mathrm{j}_o,m,\mathrm{i},r} - \theta_{\mathrm{j}_d,m,\mathrm{i},r}), \\
& \forall l\in\mathbb{L}, \quad m=1,\dots,M, \quad r=1,\dots,R_\mathrm{i}, \quad \mathrm{i}=1,\dots,N.
\end{aligned} \tag{4.24}
$$

The flow on each transmission line can not exceed its capacity:

$$
\begin{aligned}
0 \le f^+_{l,m,\mathrm{i},r} \le KL_l, \\
0 \le f^-_{l,m,\mathrm{i},r} \le KL_l,
\end{aligned} \quad \forall l\in\mathbb{L}, \quad m=1,\dots,M, \quad r=1,\dots,R_\mathrm{i}, \quad \mathrm{i}=1,\dots,N, \tag{4.25}
$$

and the phase angle and the electricity outage at each substation have to satisfy:

$$
\begin{aligned}
0 \le \theta_{\mathrm{j},m,\mathrm{i},r} &\le 2\pi, \\
o_{\mathrm{j},m,\mathrm{i},r} &\ge 0,
\end{aligned} \quad \forall \mathrm{j}\in\mathbb{J}, \quad m=1,\dots,M, \quad r=1,\dots,R_\mathrm{i}, \quad \mathrm{i}=1,\dots,N. \tag{4.26}
$$

In spite of the operational coupling constraints, they are yearly independent, and hence are detailed level decisions.

4.4.3 A Co-optimization Model

Co-optimizing the long-term capacity expansion planning with the short-term sub-hourly economic dispatch of generation and transmission leads to the following multi-scale, multi-stage stochastic program:

$$
\begin{aligned}
\text{minimize} \quad & TC^{CE} + TC^{ED} \\
\text{subject to} \quad & (4.16)-(4.18), \quad (4.20)-(4.26)
\end{aligned}
$$

$$\sum_{s=1}^{S} K_s \mathbf{k}_g^s + \sum_{\mathrm{i}=1}^{N} \sum_{r=1}^{R_{\mathrm{i}}} K_{\mathrm{i},r}^{nr} k_{g,\mathrm{i},r}^{nr} = \mathbf{0}, \quad \forall g \in \mathbb{G}, \tag{4.27}$$

where $\mathbf{k}_g^s := (k_{g,1}^s \cdots k_{g,T}^s)^T$ for all $s = 1, \dots, S$. The last equality constraint is the extended nonanticipativity constraint we proposed in Section 4.2.3. Problem (4.27) is of the same form as (4.6), and hence can be solved by the simplified N-block PCPM algorithm, proposed in Algorithm 5. Moreover, we observe that the constraint (4.17) can be removed for the decision variables $k_{g,t}^s$'s and added to the node decision variables $k_{g,\mathrm{i},r}^{nr}$'s as follows:

$$\begin{aligned} &\sum_{\mathrm{j} \in \mathbb{J}: \mathcal{J}(\mathrm{j}) = \hat{\mathrm{j}}} \; \sum_{g \in \mathbb{G}: \mathcal{G}(g) = \mathrm{j}} DF_g k_{g,\mathrm{i},r}^{nr} \geq (1 + RM_{\hat{\mathrm{j}}}) PK_{\hat{\mathrm{j}},\mathrm{i}}, \\ &\forall \hat{\mathrm{j}} \in \hat{\mathbb{J}}, \quad r = 1, \dots, R_{\mathrm{i}}, \quad \mathrm{i} = 1, \dots, N. \end{aligned} \tag{4.28}$$

Minimizing TC^{CE} subject to (4.16) and (4.18) can then be decomposed by both power generator $g \in \mathbb{G}$ and scenario $s = 1, \dots, S$. Minimizing TC^{ED} subject to (4.20) to (4.26), as well as (4.28), can then be decomposed by both realization $r = 1, \dots, R_{\mathrm{i}}$ and node $\mathrm{i} = 1, \dots, N$.

4.5 Numerical Experiments

4.5.1 Data

We consider a planning horizon of $T = 7$ years, as well as a discount factor $\delta = \frac{1}{1.02}$. For each of 12 months in a year, we consider 1 representative day, which can be further divided in 24 hours or 288 5-minutes. Hence, the total number of hours in a year is: $H = 12 \times 24 = 288$, and the total number of sub-hours (5-minutes) in a year is $M = 12 \times 288 = 3456$.

We focus on a single reserve margin region with $RM = 0.15$. In this region, we consider: 17 substations, 11 generators, 1 wind resource, 14 demand nodes and 25 transmission lines, which form a double-loop network plotted in Fig 4.5. To generate the network, we use a synthetic data set of Texas network[1] for reference.

First, we partially select 17 substations from the north central area, as well as 11 power generators, 14 demand nodes and 25 transmission lines, from the original data set. Then,

[1]The data can be found at https://electricgrids.engr.tamu.edu/.

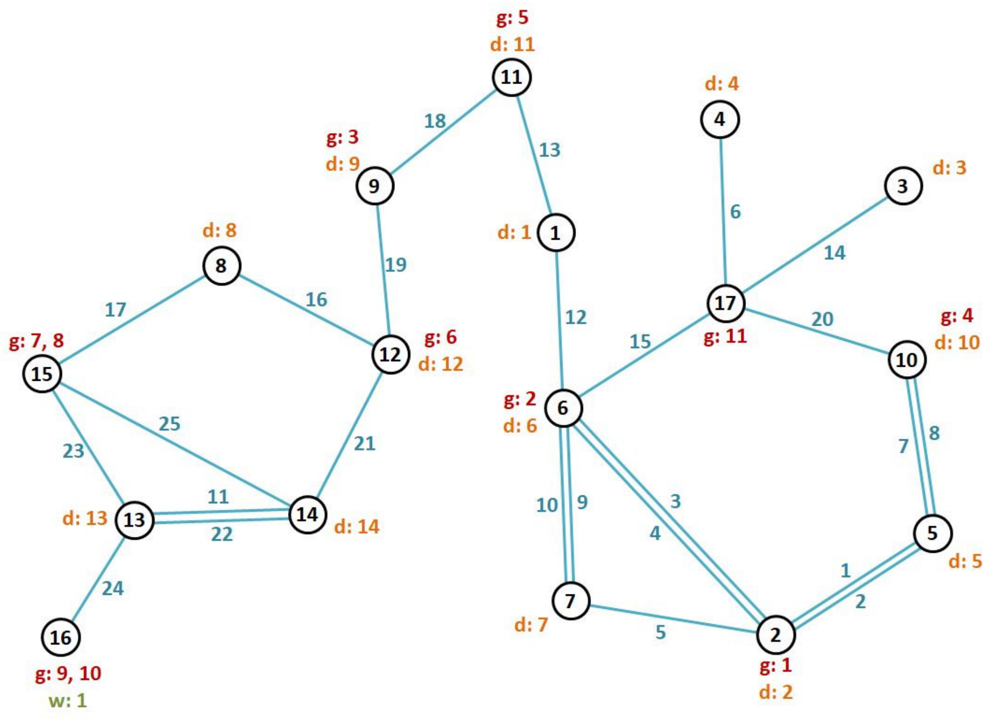


Figure 4.5. A double-loop Network in a Reserve Margin Region.

we manually assign a technology type to each power generator, with parameters set based on the historical data of EIA Annual Energy Outlook (EIA AEO)[2]. The detailed values of all parameters for the power generators can be found in Table 4.9. A single wind resource is manually added to Substation 16 with a capacity of $KW = 8.0MW$. To each demand node d in the network, we assign a load fraction LF_d, denoting the ratio of the hourly load at node d to the total hourly load of the whole north central area of Texas, listed in Table 4.10. All detailed values of the parameters for the transmission lines can be found in Table 4.11.

4.5.2 Scenario Tree Generation

The way of generating a scenario tree, presented in this subsection, is only to illustrate the numerical performance of the proposed algorithm, which will be shown in the next subsection. For more thorough studies on capacity expansion planning for a particular region, we will be using a similar method as in [47], where the stochastic process is modeled

[2]The data can be found at https://www.eia.gov/outlooks/aeo/.

Table 4.9. Data of Generation Sectors.

g	**Resp.**	j	**Type**	KG_g	N_g	FOM_g	VOM_g	RU_g	RD_g	DF_g
1	fast	2	Conventional CT	20.0	30	7.34	0.01545	100%	100%	
2	fast	6	Concentional CC	40.0	30	13.17	0.0036	33.33%	33.33%	
3	fast	9	Advanced CC	30.0	30	15.37	0.00327	50%	50%	
4	fast	10	Advanced CT	10.0	30	7.04	0.01037	100%	100%	
5	fast	11	Advanced CC	20.0	30	15.37	0.00327	50%	50%	
6	fast	12	Advanced CT	10.0	30	7.04	0.01037	100%	100%	1.0
7	slow	15	Single Advanced PC	40.0	40	37.8	0.00447	0	25%	
8	slow	15	Dual Advanced PC	60.0	40	31.18	0.00447	0	25%	
9	slow	16	Single Advanced PC	40.0	40	37.8	0.00447	0	25%	
10	slow	16	Dual Advanced PC	60.0	40	31.18	0.00447	0	25%	
11	slow	17	Nuclear	180.0	40	93.28	0.00214	0	25%	

Table 4.10. Data of Demand Nodes.

d	j	LF_d	d	j	LF_d
1	1	0.00341506840912519	8	8	0.00154893798556277
2	2	0.0101280540133409	9	9	0.00261269311300064
3	3	0.003343230400519	10	10	0.00354824502507975
4	4	0.00296193635483998	11	11	0.000419423450246929
5	6	0.000911237509166254	12	12	0.000151412418139208
6	6	0.0000204462024494551	13	13	0.0000569178068187533
7	7	0.00826579179024186	14	14	0.000143123417146185
		$\sum_{d=1}^{14} LF_d$			0.0375265178956769

Table 4.11. Data of Transmission Lines.

l	j$_o$	j$_d$	B_l	Δ_l	KL_l	l	j$_o$	j$_d$	B_l	Δ_l	KL_l
1	2	5	0.287%	126.524	150.0	14	3	17	13.934%	103.7115	1327.0
2	2	5	0.287%	126.524	150.0	15	6	17	31.84%	43.83385	1327.0
3	2	6	0.833%	43.62298	150.0	16	8	12	28.678%	64.59031	1327.0
4	2	6	0.833%	43.62298	150.0	17	8	15	30.816%	59.91346	1327.0
5	2	7	14.845%	201.1213	310.0	18	9	11	24.93%	56.07047	1327.0
6	4	17	0.683%	50.40032	150.0	19	9	12	45.045%	40.93052	1327.0
7	5	10	0.29%	125.1586	150.0	20	10	17	31.441%	44.39812	1327.0
8	5	10	0.29%	125.1586	150.0	21	12	14	58.379%	24.80828	1327.0
9	6	7	0.416%	82.76678	150.0	22	13	14	38.068%	36.6839	1327.0
10	6	7	0.416%	82.76678	150.0	23	13	15	12.077%	115.5839	1327.0
11	13	14	3.753%	6.12558	221.0	24	13	16	10.063%	119.5711	1494.0
12	1	6	22.561%	64.15568	1327.0	25	14	15	44.721%	31.2298	1327.0
13	1	11	9.567%	193.4949	1327.0						

as a geometric browning motion with the parameters verified by the historical data. It will be one of our future research tasks.

We consider a 7-level binary tree, where each node at level $t = 1, \dots, 6$ has only 2 child nodes, so there are $S = 64$ scenarios and $N = 127$ nodes in total. With each node $\mathrm{i} = 1, \dots, 127$, we associate the data vectors: $\vec{IC}_{\mathrm{i}}$, PK_{i}^{NC}, $\vec{D}_{\mathrm{i}}^{NC,WKD}$ and $\vec{D}_{\mathrm{i}}^{NC,WKE}$, where each component of $\vec{IC}_{\mathrm{i}} := (IC_{1,\mathrm{i}} \cdots IC_{11,\mathrm{i}})^T$ denotes the quadratic coefficient $IC_{g,\mathrm{i}}$ for each generation sector g at node i, PK_{i}^{NC} denotes the peak level of hourly load in the north central area at node i, each component of $\vec{D}_{\mathrm{i}}^{NC,WKD} := (D_{1,\mathrm{i}}^{NC,WKD} \cdots D_{12,\mathrm{i}}^{NC,WKD})^T$ denotes the average weekday load $D_{mon,\mathrm{i}}^{NC,WKD}$ of month $mon = 1, \dots, 12$ in the north central area at node i, and each component of $\vec{D}_{\mathrm{i}}^{NC,WKE} := (D_{1,\mathrm{i}}^{NC,WKE} \cdots D_{12,\mathrm{i}}^{NC,WKE})^T$ denotes the average weekend load $D_{mon,\mathrm{i}}^{NC,WKD}$ of month mon in the north central area at node i.

The probability of creating each child node i_c from its ancestor node i at level $t = 1, \dots, T-1$, denoted by $p_{\mathrm{i},\mathrm{i}c}$, has 2 different values, listed in Table 4.12. We consider the year of 2019 as the beginning of the planning horizon. At the root node 1, with probability $p_1 = 1.0$, the values of $\vec{IC}_1$ is obtained from the historical data of EIA AEO, and the value of PK_1^{NC} is obtained by ERCOT hourly load data from the year of 2019, as well as the values of $\vec{D}_1^{NC,WKD}$ and $\vec{D}_1^{NC,WKE}$. Each child node i_c is created with a probability $p_{\mathrm{i}_c} = p_{\mathrm{i}} \times p_{\mathrm{i},\mathrm{i}_c}$. The values of the data associated with each child node i_c is calculated using the values associated with its ancestor node times a ratio, also listed in Table 4.12. The ratio $\frac{IC_{g,\mathrm{i}_c}}{IC_{g,\mathrm{i}}}$ at each child node i_c is manually assigned. The ratio $\frac{PK_{\mathrm{i}}^{NC}}{PK_{\mathrm{i}_c}^{NC}}$ at each child node i_c is generated from the ERCOT hourly load data for the year of 2002 to 2019, as well as the ratios $\frac{\vec{D}_{\mathrm{i}_c}^{NC,WKD}}{\vec{D}_{\mathrm{i}}^{NC,WKD}}$ and $\frac{\vec{D}_{\mathrm{i}_c}^{NC,WKE}}{\vec{D}_{\mathrm{i}}^{NC,WKE}}$ at each child node i_c. Then, a 7-level binary tree can be generated. The values of $IC_{g,t}^s$ can be found at the corresponding node, and $PK_{\mathrm{i}} = \left(\sum_{d=1}^{14} LF_d\right) \times PK_{\mathrm{i}}^{NC}$.

In our model, we also consider an uncertainty for the sub-hourly load and the wind availability factor. At each node $\mathrm{i} = 1, \dots, 127$, we consider 10 realizations with a equal probability $p_r^{rl} = 0.1$. For each month $mon = 1, \dots, 12$, two data pools of 5-minute load portions to a daily load are made for weekdays and weekend days respectively, using the historical data of ERCOT. For each realization r, a sequence of 12 representative days in each month is sampled. For each representative day in each month $mon = 1, \dots, 12$, a sequence of total 288 5-minutes load portions in a day is drawn from a data pool, depending

Table 4.12. Probabilities and Ratios for Generating the Binary Scenario Tree.

	Decreasing		**Increasing**			
	$p_{\mathrm{i},\mathrm{i}_c}$				p_1	
	0.411765		0.588235		1.0	
g	$\frac{IC_{g,\mathrm{i}_c}}{IC_{g,\mathrm{i}}}$				$IC_{g,1}$	
1	0.999		1.07		4.87	
2	0.999		1.06		1.48	
3	0.999		1.04		2.56	
4	0.999		1.05		3.22	
5	0.999		1.04		2.56	
6	0.999		1.05		3.22	
7	0.999		1.08		4.99	
8	0.999		1.09		2.26	
9	0.999		1.08		4.99	
10	0.999		1.09		2.26	
11	1.0		1.0		2.48	
	$\frac{PK_{\mathrm{i}_c}^{NC}}{PK_{\mathrm{i}}^{NC}}$				PK_1^{NC}	
	0.962635		1.047867		25493.791364	
mon	$\frac{D_{mon,\mathrm{i}_c}^{NC,WKD}}{D_{mon,\mathrm{i}}^{NC,WKD}}$	$\frac{D_{mon,\mathrm{i}_c}^{NC,WKE}}{D_{mon,\mathrm{i}}^{NC,WKE}}$	$\frac{D_{mon,\mathrm{i}_c}^{NC,WKD}}{D_{mon,\mathrm{i}}^{NC,WKD}}$	$\frac{D_{mon,\mathrm{i}_c}^{NC,WKE}}{D_{mon,\mathrm{i}}^{NC,WKE}}$	$D_{mon,1}^{NC,WKD}$	$D_{mon,1}^{NC,WKE}$
1	0.939811	0.922987	1.079484	1.095487	328060.997010	314257.280135
2	0.922397	0.933038	1.087484	1.086450	312898.959607	301457.547991
3	0.976826	0.964011	1.042382	1.049035	286408.897311	273070.060138
4	0.965699	0.946616	1.037831	1.055578	270477.852282	250265.200286
5	0.948244	0.928000	1.059238	1.075360	313919.847511	285299.536478
6	0.940542	0.937028	1.058858	1.073340	360777.718191	352160.898949
7	0.941563	0.933001	1.065955	1.072776	407644.239217	391658.771474
8	0.946417	0.936104	1.064896	1.072610	446160.028895	412306.627903
9	0.942981	0.928792	1.074786	1.092140	409638.432070	394556.334726
10	0.970548	0.953333	1.047375	1.062817	305789.782162	283395.415091
11	0.974836	0.966007	1.045072	1.052684	295388.955169	268610.114623
12	0.951374	0.936434	1.058845	1.073102	308500.818307	276917.790869

on the day type. Then, $D_{d,m,\mathrm{i},r}$ is calculated by multiplying $D_{mon,\mathrm{i}}^{NC,WKD}$ or $D_{mon,\mathrm{i}}^{NC,WKE}$, whose month mon contains the sub-hour m, with the portion in sub-hour m and the load fraction LF_d. Similarly, for each representative day in each month $mon = 1, \ldots, 12$, a sequence of 288 average wind availability factor during 5 minutes in a day is drawn from a data pool corresponding to month *mon*, made by the historical data of WIND Toolkit from NREL[3]. The whole sequence of 12 months determines the values of $a_{m,\mathrm{i},r}$.

4.5.3 Numerical Results

We compare the performance of the simplified PCPM algorithm with that of the ADMM algorithm and the PHA algorithm, shown in Table 4.13. For both simplified PCPM and

Table 4.13. Numerical Results for Sub-hourly Modeling on a multi-node computer cluster.

Algorithm	**Simplified PCPM**	**ADMM**	**PHA**
decomposed by	scenario and node-realization	scenario and node-realization	scenario only
total number of decomposed sub-problems	1334	1334	64
largest size of decision variable in each decomposed sub-problem	$312,491$	$312,491$	$21,873,754$
largest number of constraints in each decomposed sub-problem	$211,417$	$211,417$	$14,799,204$
total number of processors	1334	1334	64
total number of nodes (maximum 20 cores per node)	67	67	64
maximum memory per node	6.8 GB	6.8 GB	20.4 GB
total elapsed wall-clock time	6.92 h	6.98 h	8.57 h
total number of iterations	1196	1228	6
τ	0.0001	0.0002	1.2150
total cost	$233,998.562$ (k$)	$234,497.601$ (k$)	$235,795.482$ (k$)
outage penalty	0.025424 (k$)	0.030819 (k$)	7.961365 (k$)

ADMM, τ measures the average residual of the extended nonanticipativity constraints; that is:

$$\tau = \frac{1}{\sqrt{n_{sc}^Y + n_{nr}^Y}} \left\| \sum_{s=1}^{S} K_s \mathbf{y}^{s,k} + \sum_{\mathrm{i}=1}^{N} \sum_{r=1}^{R_\mathrm{i}} K_{\mathrm{i},r}^{nr} \mathbf{y}_{\mathrm{i},r}^{nr,k} \right\|_2 .$$

For PHA, τ measures the average residual of the original nonanticipativity constraints; that is: $\tau = \frac{1}{\sqrt{n_{sc}}} \left\| \sum_{s=1}^{S} K_s \mathbf{x}^{s,k} \right\|_2$.

Compared with PHA, simplified PCPM, as well as ADMM, decomposes the large-scale problem by both scenario and node-realization, while PHA only decomposes the problem

[3]The data can be found at https://www.nrel.gov/grid/wind-toolkit.html.

by scenario. Using the hybrid decomposition method, the original problem is decomposed into 1334 sub-problems, whose largest size of the decision variable is approximately 3×10^5, with a largest number of constraints being around 2×10^5. However, using only scenario decomposition, the problem can only be decomposed into 64 sub-problems, causing a fact that both the size of the decision variable and the number of constraints are almost 70 times of that in the hybrid decomposition. A much larger-sized sub-problem not only requires more amount of memory but also takes more time to be solved in each iteration.

Within a 9-hour usage of computing resources on a multi-node computer cluster, PHA is implemented on 64 processors, each of which corresponds to a computing unit that solves a sub-problem. The 64 processors are mapped to 64 nodes with 1 processor per node. The memory usage on each node is about 20 GB. The algorithm terminates with a much larger average residual, as well as much higher total cost and outage penalty, compared with simplified PCPM and PHA.

Both simplified PCPM and ADMM are implemented on 1334 processors, each of which solves a decomposed sub-problem. The 1334 processors are mapped to 67 nodes with maximum 20 cores per node. Compared with PHA, both two algorithms use much less memory per node, which demonstrates the strengths of using hybrid decomposition. Additionally, compared with ADMM, simplified PCPM converges with fewer number of iterations and less elapsed wall-clock time, due to the benefits of exploiting the technique of orthogonal projection. The algorithm also terminates with a smaller average residual, as well as lower total cost and outage penalty.

4.6 Conclusion and Future Works

In this chapter, we apply the N-block PCPM algorithm to solve multi-scale multi-stage stochastic programs, with the application to electricity capacity expansion models. Numerical results show that the proposed simplified N-block PCPM algorithm, along with the the hybrid decomposition method, exhibits much better scalability for solving the resulting deterministic, large-scale block-separable optimization problem, when compared with the ADMM algorithm and the PHA algorithm. The superiority of algorithm's scalability is attributed to the two key features of the algorithm design: first, the proposed hybrid scenario-node-realization decomposition method with extended nonanticipativity constraints can decompse the original problem under various uncertainties of different temporal scales; second, when applying the N-block PCPM algorithm to solve the resulting deterministic, large-scale N-block convex optimization problem, the technique of orthogonal projection we exploit greatly simplifies the iteration steps and reduce the communication overhead among all computing units, which also contributes to the efficiency of the algorithm.

Numerical experiments with better ways of scenario generation will be conducted in the future. Retirement of generators, as well as storage of electricity, will be considered in future models. The number of substations will also be increased, when more computing resources become available.

5. CONCLUDING REMARKS

5.1 Summary

This dissertation develops efficient and scalable distributed algorithms for solving large-scale constrained convex optimization problems, with global convergence established. First, an extended N-block PCPM algorithm is proposed to solve N-block convex optimization problems with not only linear but also nonlinear coupling constrains, which cannot be directly handled by ADMM-type or PCPM-type algorithms. Both sub-linear and linear convergence rates are proved, under different conditions, by the numerical experiments. Secondly, an asynchronous N-block PCPM algorithm is proposed to solve linearly constrained N-block convex optimization problems, as a starting point, with a standard bounded delay assumption. The proposed algorithm is applied to solve a graph optimization problem arising from spatial clustering. A global sub-linear convergence rate is proved, with additional assumption of strong convexity of the objective function. The efficiency of using asynchronous iterations is demonstrated by the numerical results. Thirdly, a Jacobi-style distributed algorithm is proposed to solve convex QCQPs, using a novel idea of predictor-corrector primal-dual update with an adaptive step size, to deal with the non-separability of both objective function and coupling constraints. Extensive numerical experiments on various large-scale data sets are conducted on a multi-node computer cluster. Numerical results show that, compared with the centralized algorithm, the proposed algorithm exhibits favourable scalability due to its amenability to massive parallel computing, as well as distributed storage of data. Finally, the N-block PCPM algorithm is applied to solve a real-world application problem of electricity capacity expansion, modeled as a multi-scale, multi-stage stochastic program. By introducing extended nonanticipativity constraints, a hybrid scenario-node-realization decomposition method is proposed, which decomposes the problem under uncertainties with different temporal scales, and thus is ready for distributed algorithms with the ability of massive parallelization. A technique of orthogonal projection is also exploited to simplify the iteration step and to reduce the communication overhead among all computing units, which leads to a scalable algorithm for solving the resulting deterministic, large-scale optimization problem.

5.2 Future Research

The algorithms proposed in this dissertation, as well as application to problems from machine learning and broad engineering areas, provide a convenient distributed framework for solving large-scale constrained convex optimization problems. Useful extensions of this work are to further enhance the ability of the distributed framework for large-scale constrained optimization. A few possible areas for future research are discussed below.

• Constrained Multi-convex Optimization Problems

While the direct extension of the proposed distributed algorithms to solve constrained non-convex optimization problems might be difficult, we can start with a constrained multi-convex optimization problem instead. More specifically, we consider the following optimization problem:

$$\begin{aligned} \underset{\mathbf{x}_1,\ldots,\mathbf{x}_N}{\text{minimize}} \quad & f(\mathbf{x}_1,\ldots,\mathbf{x}_N) \\ \text{subject to} \quad & (\mathbf{x}_1,\ldots,\mathbf{x}_N) \in \mathcal{X}, \end{aligned} \tag{5.1}$$

where each $\mathbf{x}_\mathrm{i} \in \mathbb{R}^{n_\mathrm{i}}$ is a block of the decision variable for all $\mathrm{i} = 1,\ldots,N$, and $n = \sum_{\mathrm{i}=1}^{N} n_\mathrm{i}$. The objective function $f : \mathbb{R}^n \to \mathbb{R}$ is *block multi-convex*; that is, for each $\mathrm{i} = 1,\ldots,N$, the function $f(\mathbf{x}_\mathrm{i};\mathbf{x}'_{-\mathrm{i}}) : \mathbb{R}^{n_\mathrm{i}} \to \mathbb{R}$ is convex, given $\mathbf{x}'_{-\mathrm{i}}$ is fixed, where $\mathbf{x}_{-\mathrm{i}}$ denotes all blocks of decision variables except for $\mathbf{x}_\mathrm{i}$. The abstract constraint set $\mathcal{X}$ is also block multi-convex; that is, given a fixed $\mathbf{x}'_{-\mathrm{i}}$, the set

$$\mathcal{X}_\mathrm{i}(\mathbf{x}_\mathrm{i};\mathbf{x}'_{-\mathrm{i}}) \coloneqq \left\{ \mathbf{x}_\mathrm{i} \in \mathbb{R}^{n_\mathrm{i}} \,\middle|\, (\mathbf{x}'_1,\ldots,\mathbf{x}'_{\mathrm{i}-1},\mathbf{x}_\mathrm{i},\mathbf{x}'_{\mathrm{i}+1},\ldots,\mathbf{x}'_N) \in \mathcal{X} \right\}$$

is convex for all $\mathrm{i} = 1,\ldots,N$. Such problems arise from many application areas, such as non-negative tensor factorization [48], and weakly-constrained multi-task Learning [49]. In [48], the authors present three types of distributed algorithms for such constrained multi-convex optimization problem and analyze convergence with either Lipschitz differentiability or assumption of strongly convexity. It is worth exploring if the (asynchronous) N-block PCPM algorithm can be extended to solve problem (5.1), and if the global (linear) convergence can be established under other possibly more mild conditions.

• Decentralized Algorithms

When applying (asynchronous) distributed algorithms to solve large-scale constrained optimization problems, other than the specific main-worker paradigm presented in Section 2.3, we can consider more general graph topology among all computing units to design communication-efficient decentralized algorithms. A consensus-based decentralized algorithms are proposed in [50], [51]. It's of great interest to develop (asynchronous) decentralized algorithms for optimization problems with general coupling constraints.

• **Cloud Computing Resources**

The use of cloud computing resources is also to be explored in the future, such as Amazon Web Services (AWS), to test the performance of the proposed distributed algorithms on more application problems with larger data sets.

REFERENCES

[1] J. Nocedal and S. Wright, *Numerical Optimization.* Springer, 2006.

[2] S. Boyd, L. Xiao, A. Mutapcic, and J. Mattingley, “Notes on decomposition methods,” *Notes for EE364B, Stanford University*, pp. 1–36, 2007.

[3] S. Boyd, N. Parikh, and E. Chu, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[4] R. T. Rockafellar, *Convex Analysis.* Princeton University Press, 2015.

[5] W. Deng and W. Yin, “On the global and linear convergence of the generalized alternating direction method of multipliers,” *Journal of Scientific Computing*, vol. 66, no. 3, pp. 889–916, 2016.

[6] M. Hong and Z.-Q. Luo, “On the linear convergence of the alternating direction method of multipliers,” *Mathematical Programming*, vol. 162, no. 1-2, pp. 165–199, 2017.

[7] G. Chen and M. Teboulle, “A proximal-based decomposition method for convex minimization problems,” *Mathematical Programming*, vol. 64, no. 1-3, pp. 81–101, 1994.

[8] P. Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer, “Accelerated gradient methods and dual decomposition in distributed model predictive control,” *Automatica*, vol. 49, no. 3, pp. 829–833, 2013.

[9] J. Dass, V. P. Sakuru, V. Sarin, and R. N. Mahapatra, “Distributed QR decomposition framework for training support vector machines,” in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2017, pp. 753–763.

[10] H. Yu and M. J. Neely, “A simple parallel algorithm with an o(1/t) convergence rate for general convex programs,” *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 759–783, 2017.

[11] D. Hallac, J. Leskovec, and S. Boyd, “Network lasso: Clustering and optimization in large graphs,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2015, pp. 387–396.

[12] W. Deng, M.-J. Lai, Z. Peng, and W. Yin, “Parallel multi-block ADMM with o (1/k) convergence,” *Journal of Scientific Computing*, vol. 71, no. 2, pp. 712–736, 2017.

[13] H. Wang, A. Banerjee, and Z.-Q. Luo, “Parallel direction method of multipliers,” in *Advances in Neural Information Processing Systems*, 2014, pp. 181–189.

[14] X. Wang, M. Hong, S. Ma, and Z.-Q. Luo, "Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers," *arXiv preprint arXiv:1308.5294*, 2013.

[15] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[16] R. T. Rockafellar, "Augmented Lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of Operations Research*, vol. 1, no. 2, pp. 97–116, 1976.

[17] J. Asaadi, "A computational comparison of some non-linear programs," *Mathematical Programming*, vol. 4, no. 1, pp. 144–154, 1973.

[18] S. Sahni and G. Vairaktarakis, "The master-slave paradigm in parallel computer and industrial settings," *Journal of Global Optimization*, vol. 9, no. 3-4, pp. 357–377, 1996.

[19] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization—part i: Algorithm and convergence analysis," *IEEE Transactions on Signal Processing*, vol. 64, no. 12, pp. 3118–3130, 2016.

[20] T.-H. Chang, W.-C. Liao, M. Hong, and X. Wang, "Asynchronous distributed admm for large-scale optimization—part ii: Linear convergence analysis and numerical performance," *IEEE Transactions on Signal Processing*, vol. 64, no. 12, pp. 3131–3144, 2016.

[21] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, no. Jan, pp. 27–72, 2004.

[22] Y. Huang and D. P. Palomar, "Randomized algorithms for optimal solutions of double-sided qcqp with applications in signal processing," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1093–1108, 2014.

[23] O. Rabaste and L. Savy, "Mismatched filter optimization for radar applications using quadratically constrained quadratic programs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 4, pp. 3107–3122, 2015.

[24] C. Aholt, S. Agarwal, and R. Thomas, "A qcqp approach to triangulation," in *European Conference on Computer Vision*, Springer, 2012, pp. 654–667.

[25] S. Bose, D. F. Gayme, K. M. Chandy, and S. H. Low, "Quadratically constrained quadratic programs on acyclic graphs with application to power flow," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 3, pp. 278–287, 2015.

[26] Y. Nesterov and A. Nemirovskii, *Interior-point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[27] A. Nemirovski, "Interior point polynomial time methods in convex programming," *Lecture Notes*, 2004.

[28] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and Its Applications*, vol. 284, no. 1-3, pp. 193–228, 1998.

[29] *IBM ILOG CPLEX optimization studio CPLEX User's Manual, Version 12 Release 7*, 1987-2017.

[30] B. O'donoghue, E. Chu, N. Parikh, and S. Boyd, "Conic optimization via operator splitting and homogeneous self-dual embedding," *Journal of Optimization Theory and Applications*, vol. 169, no. 3, pp. 1042–1068, 2016.

[31] A. Kalbat and J. Lavaei, "A fast distributed algorithm for decomposable semidefinite programs," in *54th IEEE Conference on Decision and Control*, 2015, pp. 1742–1749.

[32] S. K. Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer, "Distributed semidefinite programming with application to large-scale system analysis," *IEEE Transactions on Automatic Control*, vol. 63, no. 4, pp. 1045–1058, 2018.

[33] K. Huang and N. D. Sidiropoulos, "Consensus-ADMM for general quadratically constrained quadratic programming," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5297–5310, 2016.

[34] K. Basu, A. Saha, and S. Chatterjee, "Large-scale quadratically constrained quadratic program via low-discrepancy sequences," in *Advances in Neural Information Processing Systems*, 2017, pp. 2297–2307.

[35] *IBM ILOG CPLEX optimization studio CPLEX Parameters Reference, Version 12 Release 8*, 1987-2017.

[36] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[37] L. Breiman *et al.*, "Arcing classifier," *The Annals of Statistics*, vol. 26, no. 3, pp. 801–849, 1998.

[38] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 2012.

[39] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 2013.

[40] P. Lancaster and H. K. Farahat, “Norms on direct sums and tensor products,” *Mathematics of Computation*, vol. 26, no. 118, pp. 401–414, 1972.

[41] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

[42] W. Römisch, “Scenario generation,” *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

[43] T. Pennanen, “Epi-convergent discretizations of multistage stochastic programs via integration quadratures,” *Mathematical Programming*, vol. 116, no. 1-2, pp. 461–479, 2009.

[44] K. Høyland and S. W. Wallace, “Generating scenario trees for multistage decision problems,” *Management Science*, vol. 47, no. 2, pp. 295–307, 2001.

[45] G. C. Pflug, “Scenario tree generation for multiperiod financial optimization by optimal discretization,” *Mathematical Programming*, vol. 89, no. 2, pp. 251–271, 2001.

[46] R. T. Rockafellar and R. J.-B. Wets, “Scenarios and policy aggregation in optimization under uncertainty,” *Mathematics of Operations Research*, vol. 16, no. 1, pp. 119–147, 1991.

[47] S. Jin, S. M. Ryan, J.-P. Watson, and D. L. Woodruff, “Modeling and solving a large-scale generation expansion planning problem under uncertainty,” *Energy Systems*, vol. 2, no. 3-4, pp. 209–242, 2011.

[48] Y. Xu and W. Yin, “A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion,” *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1758–1789, 2013.

[49] J. Wang, L. Zhao, and L. Wu, “Multi-convex inequality-constrained alternating direction method of multipliers,” *arXiv preprint arXiv:1902.10882*, 2019.

[50] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, “On the linear convergence of the ADMM in decentralized consensus optimization,” *IEEE Transactions on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[51] W. Shi, Q. Ling, G. Wu, and W. Yin, “Extra: An exact first-order algorithm for decentralized consensus optimization,” *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.