RICE UNIVERSITY

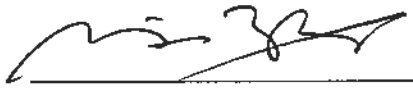
Generalizations of the Alternating Direction Method of Multipliers for Large-Scale and Distributed Optimization

by

Wei Deng

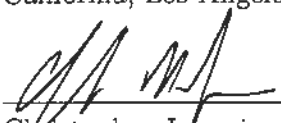
A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree
Doctor of Philosophy

Approved, Thesis Committee:

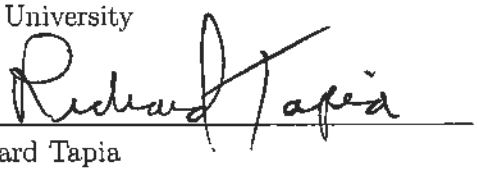


Yin Zhang, Chair
Professor of Computational and Applied Mathematics, Rice University

Wotao Yin, Thesis Director
Professor of Mathematics, University of California, Los Angeles



Christopher Jermaine
Associate Professor of Computer Science, Rice University



Richard Tapia
University Professor
Maxfield-Oshman Professor in Engineering
Professor of Computational and Applied Mathematics, Rice University

Houston, Texas
May, 2014

ABSTRACT

Generalizations of the Alternating Direction Method of Multipliers for Large-Scale and Distributed Optimization

by

Wei Deng

The alternating direction method of multipliers (ADMM) has been revived in recent years due to its effectiveness at solving many large-scale and distributed optimization problems, particularly arising from the areas of compressive sensing, signal and image processing, machine learning and statistics. This thesis makes important generalizations to ADMM as well as extending its convergence theory.

We propose a generalized ADMM framework that allows more options of solving the subproblems, either exactly or approximately. Such generalization is of great practical importance because it brings more flexibility in solving the subproblems efficiently, thereby making the entire algorithm run faster and scale better for large problems. We establish its global convergence and further show its linear convergence under a variety of scenarios, which cover a wide range of applications. The derived rate of convergence also provides some theoretical guidance for optimizing the parameters of the algorithm. In addition, we introduce a simple technique to improve an existing convergence rate from O(1/k) to o(1/k).

Moreover, we introduce a parallel and multi-block extension to ADMM for solving convex separable problems with multiple blocks of variables. The algorithm decomposes the original problem into smaller subproblems and solves them in parallel at

each iteration. It can be implemented in a fully distributed manner and is particularly attractive for solving certain large-scale problems. We show that extending ADMM straightforwardly from the classic Gauss-Seidel setting to the Jacobi setting, from two blocks to multiple blocks, will preserve convergence if the constraint coefficient matrices are mutually near-orthogonal and have full column-rank. For general cases, we propose to add proximal terms of different kinds to the subproblems, so that they can be solved in flexible and efficient ways and the algorithm converges globally at a rate of o(1/k). We introduce a strategy for dynamically tuning the parameters of the algorithm, often leading to faster convergence in practice. Numerical results are presented to demonstrate the efficiency of the proposed algorithm in comparison with several existing parallel algorithms. We also implemented our algorithm on Amazon EC2, an on-demand public computing cloud, and report its performance on very large-scale basis pursuit problems with distributed data.

Acknowledgements

First of all, I would like to express my utmost gratitude to my thesis advisors, Dr. Wotao Yin and Dr. Yin Zhang, for their continuous guidance, support and encouragement throughout my Ph.D. career. They are great educators and mentors, who introduced me to this exciting field of numerical optimization and helped me develop critical thinking and research skills. During the preparation and completion of this study, they have offered me a countless number of invaluable suggestions and inspirational ideas. Without them, this work would not have been possible.

My gratitude also goes to my thesis committee members, Dr. Christopher Jermaine and Dr. Richard Tapia. I am thankful for their valuable questions and comments that helped improve my thesis.

My research has benefited from the collaboration and discussions with Dr. Ming-Jun Lai and Zhimin Peng. I would like to thank them for carefully reading my manuscript and giving me many constructive suggestions. I am also very grateful to many colleagues and friends who provide me helpful feedback and encouragement on my research work.

I wish to thank all the faculty members, administrative staff and fellow students of the department for providing me with an excellent environment and atmosphere for study and research.

I am deeply indebted to my family for all their never-ending love, encouragement, and support at all times. I could never thank my parents enough for always caring about me and believing in me. Special thanks to my wife Christina and my daughter Emilia for bringing me such a wonderful life. To them, I dedicate this thesis.

My research is supported in part by NSF Grants NSF DMS-1349855 and ECCS-1028790, and ARO MURI W911NF-09-1-0383.

Contents

Illustrations

Tables

Chapter 1

Introduction

With the rapid advancement of modern technology, enormous amount of data is being generated in diverse areas such as the Internet, mobile devices, cameras, business, research and engineering. There is a dramatically increasing demand for processing and interpreting the often extremely large datasets. However, the size and complexity of the modern datasets have been growing beyond the ability of the traditional methods, and have become a major bottleneck for many applications. This phenomenon is often referred to as the "Big Data". Therefore, efficient and scalable methods are highly desirable to cope with the size and complexity of the modern datasets.

This thesis focuses on a particular optimization method – the alternating direction method of multipliers (ADMM), which is well suited to many of the Big Data applications. ADMM uses a "divide and conquer" strategy to decompose an often large and difficult problem into smaller and easier subproblems, which can be processed efficiently and in parallel. The method can be implemented in a fully decentralized manner to process distributed and huge datasets. Due to these favorable features, this classic optimization method has experienced a renaissance in the recent decade. As a versatile algorithmic tool, ADMM has proven to be very effective at solving many large-scale and distributed optimization problems, particularly arising from the areas of compressive sensing, signal and image processing, machine learning and applied statistics.

In this thesis, we generalize the classic ADMM in various ways, aiming to further

improve its efficiency, flexibility and applicability for large-scale and distributed optimization problems. We also make important extensions to the convergence theory and establish better convergence rates. The structure of this thesis is organized as follows:

- In Chapter 2, we briefly review ADMM and the related literature. We give an overview of several important applications which motivate the study of this thesis. In addition, we introduce some basic mathematical notation and preliminary knowledge in convex analysis that will be used frequently in the later chapters.

- In Chapter 3, we introduce a generalized ADMM framework that allows more options of solving the subproblems either exactly or approximately, such as linearizing the subproblems, taking one gradient descent step, and approximating the Hessian. The generalization is of great practical importance because it brings more flexibility in solving the subproblems efficiently, thereby making the entire algorithm run faster and scale better for large problems. Furthermore, we establish the global convergence of the generalized ADMM under some standard assumptions. In addition, we give a brief literature review on the convergence rates of ADMM. We introduce a simple technique to improve an existing $O(1/k)$ convergence rate to $o(1/k)$, where k is the number of iterations.

- In Chapter 4, we mainly establish the global linear convergence of the generalized ADMM under a variety of scenarios. Among these scenarios, we require that at least one of the two objective functions is strictly convex and has Lipschitz continuous gradient, along with certain full rank conditions on the constraint coefficient matrices. These scenarios commonly arise in many ap-

plications and we will give several examples. In addition, we carry out simple numerical experiments to demonstrate the linear convergence of ADMM on the *elastic net* and *distributed Lasso* problems.

- In Chapter 5, we introduce a parallel and multi-block extension to ADMM, which is well suited to distributed computing and is particularly attractive for solving certain large-scale problems. We show that extending ADMM straightforwardly from the classic Gauss-Seidel setting to the Jacobi setting, from 2 blocks to N blocks, will preserve convergence if the constraint coefficient matrices are mutually near-orthogonal and have full column-rank. Without these assumptions, this straightforward extension may fail to converge in general. Therefore, we propose a simple modification by adding proximal terms of different kinds to the subproblems. We show that this modification not only allows more flexible and efficient ways of solving the subproblems, but more importantly makes the algorithm converges globally at a rate of $o(1/k)$. In addition, we introduce a strategy for dynamically tuning the parameters of the algorithm, often leading to faster convergence in practice. We present numerical results to demonstrate the efficiency of the proposed algorithm in comparison with several existing parallel algorithms. We also implemented our algorithm on Amazon EC2, an on-demand public computing cloud, and report its performance on very large-scale *basis pursuit* problems with distributed data.

- In Chapter 6, we summarize and conclude the thesis.

Chapter 2

Background and Preliminary

In this chapter, we provide some background and preliminary knowledge necessary to understand the later chapters of the thesis. First, in Section 2.1, we give a brief introduction to the augmented Lagrangian method, a precursor to the alternating direction method of multipliers. Then in Section 2.2, we review the alternating direction method of multipliers and the related literature. In Section 2.3, we mention several important applications which motivate the study of this thesis. In Section 2.4, we introduce some basic mathematical notation and preliminary knowledge in convex analysis that will be used frequently in the later chapters.

2.1 Augmented Lagrangian Method

The classic *augmented Lagrangian method*, also known as the *method of multipliers*, was first introduced by [37,49] in the late 1960s and has been a popular methodology for solving constrained optimization problems. Let us consider the canonical convex optimization problem with linear constraints:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & f(x) \\ \text{s.t. } & Ax = b, \end{aligned} \tag{2.1}$$

where $x \in \mathbb{R}^n$ is an unknown variable, $A \in \mathbb{R}^{p\times n}$ and $b \in \mathbb{R}^p$ are given, and $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is an extended-value convex function. Note that f is allowed to take the value $+\infty$ so that any constraints on x can be "embedded" in f. That is, if

x is constrained in some closed convex set $\mathcal{X} \subseteq \mathbb{R}^n$, then it is equivalent to include the indicator functions $I_{\mathcal{X}}(x)$ as part of the objective function f, where the indicator function of a convex set $\mathcal{X}$ is defined by

$$I_{\mathcal{X}}(x) := \begin{cases} 0 & \text{if } x \in \mathcal{X}, \\ +\infty & \text{if } x \notin \mathcal{X}. \end{cases} \tag{2.2}$$

The so-called *augmented Lagrangian* of (2.1) is given by

$$\mathcal{L}_\beta(x, \lambda) := f(x) - \lambda^T(Ax - b) + \frac{\beta}{2}\|Ax - b\|_2^2, \tag{2.3}$$

where $\lambda \in \mathbb{R}^p$ is the Lagrangian multiplier (or dual variable), and $\beta > 0$ is a penalty parameter. It can be viewed as a combination of the standard Lagrangian function and a quadratic penalty term on the constraints.

The augmented Lagrangian method is an iterative algorithmic framework that solves a sequence of unconstrained subproblems based on the augmented Lagrangian function. It starts with an initial estimate λ^0 to the Lagrangian multiplier and iterates as follows: for $k = 0, 1, \ldots,$

$$x^{k+1} = \arg\min_x \ \mathcal{L}_\beta(x, \lambda^k) \tag{2.4}$$

$$\lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} - b) \tag{2.5}$$

At each iteration, it minimizes the augmented Lagrangian over the primal variable x by fixing the dual variable λ, and then makes a simple update to λ using the updated x. In addition, the fixed penalty parameter β can also be replaced by a sequence of variable parameters $\{\beta^k\}$ updated at every iteration.

The augmented Lagrangian method is closely related to the *dual ascent method* (or *dual subgradient method*) [53] as well as the *quadratic penalty method*. Compared to these methods, the augmented Lagrangian method is more robust and converges under

much milder conditions. Unlike the quadratic penalty method where the penalty parameter needs to go to infinity, the penalty parameter β of the augmented Lagrangian method can be fixed and stay much smaller, thereby avoiding ill-conditioning of the subproblems.

2.2 Alternating Direction Method of Multipliers

The alternating direction method of multipliers (ADMM) is a variant of the augmented Lagrangian method, which is applied to constrained optimization problems with *separable* objective functions in the following form:

$$\begin{aligned} \min_{x,y} \ & f(x) + g(y) \\ \text{s.t. } & Ax + By = b, \end{aligned} \tag{2.6}$$

where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ are unknown variables, $A \in \mathbb{R}^{p \times n}$ and $B \in \mathbb{R}^{p \times m}$ are given matrices, $b \in \mathbb{R}^p$ is a given vector, and $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ are extended-value convex functions. Constraints $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ are closed convex sets, can be included as the indicator functions $I_{\mathcal{X}}(x)$ and $I_{\mathcal{Y}}(y)$ in the objective functions f and g, respectively.

Applying the augmented Lagrangian method to (2.6) yields the following iterations:

$$(x^{k+1}, y^{k+1}) = \arg\min_{x,y} \ \mathcal{L}_\beta(x, y, \lambda^k), \tag{2.7}$$

$$\lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - b). \tag{2.8}$$

Here, the augmented Lagrangian of (2.6) is given by:

$$\mathcal{L}_\beta(x, y, \lambda) = f(x) + g(y) - \lambda^T(Ax + By - b) + \frac{\beta}{2}\|Ax + By - b\|_2^2, \tag{2.9}$$

where $\lambda \in \mathbb{R}^p$ is the Lagrangian multiplier and $\beta > 0$ is a penalty parameter. Note that though x and y are separable in the objective functions, they are coupled in the augmented Lagrangian through the quadratic penalty term.

In many applications, it is often relatively easy to minimize the functions f and g individually; however, minimizing both of f and g at the same time is much more difficult. Therefore, the joint minimization step (2.7) of the augmented Lagrangian method often becomes a bottleneck for these applications. In contrast, the alternating direction method of multipliers (see Algorithm 1 below) replaces the joint minimization step by two "alternating minimization" steps in which $\mathcal{L}_\beta(x, y, \lambda^k)$ in (2.7) is minimized over x and y separately, one after another. We refer to the step 3 and step 4 of Algorithm 1 as y-subproblem and x-subproblem of ADMM, respectively.

Algorithm 1: Classic ADMM

1 Initialize $x^0,\ \lambda^0,\ \beta > 0$;

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

3 $\quad y^{k+1} = \arg\min_y \mathcal{L}_\beta(x^k, y, \lambda^k)$;

4 $\quad x^{k+1} = \arg\min_x \mathcal{L}_\beta(x, y^{k+1}, \lambda^k)$;

5 $\quad \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - b)$.

Compared to the augmented Lagrangian method, ADMM can take advantage of the separability of the problem (2.6) and thus is particularly more effective on this kind of problems. ADMM may take more iterations to converge, due to less accurate minimization of the augmented Lagrangian $\mathcal{L}_\beta(x, y, \lambda^k)$ than (2.7) at each iteration. However, ADMM often runs faster due to the much cheaper subproblems.

ADMM was first introduced in the 1970s by [21,23] with the application in solving partial differential equations. Its root traces back to the Douglas-Rachford splitting

method [15] in the 1950s. However, this classic optimization method was not widely known until the recent past. In the last few years, ADMM has experienced a significant revival and has found numerous successful applications, particularly in the areas of compressive sensing, signal and image processing, machine learning, applied statistics and operations research. We refer to [4, 6, 14, 18, 27, 29, 39, 42, 59, 61] for a number of recent applications. With the relative ease of implementation, ADMM often leads to state-of-the-art performance on many large-scale problems. Also, ADMM is well suited to parallel and distributed computing. It can be implemented in a fully decentralized computational environment, with the scalability to process very huge datasets. These benefits have mainly contributed to the renaissance of ADMM and are of great importance to modern applications with extremely large datasets.

2.3 Applications

The alternating direction method of multipliers has wide applications in diverse areas. In this section, we review several important applications as motivating examples.

2.3.1 Convex Regularization

Convex regularization models have been widely used in various applications:

$$\min_x f(Ax - b) + g(x) \tag{2.10}$$

where $x \in \mathbb{R}^n$ is unknown variable, $A \in \mathbb{R}^{p\times n}$ and $b \in \mathbb{R}^p$ are given data, and f and g are convex functions. Here, f is referred to as the *loss* (or *data fidelity*) function for penalizing data-fitting errors; g is the *regularization* function in order to prevent over-fitting or promote certain structures in the solutions. For example, the loss function f can be chosen as squared ℓ_2-norm $\|\cdot\|_2^2$ (least squares), ℓ_1-norm $\|\cdot\|_1$ (least

absolute deviations), Huber function, indicator function of a closed convex set, and so on. Several commonly used regularization functions g include squared ℓ_2-norm $\|\cdot\|_2^2$, total variation, ℓ_1-norm, nuclear norm, just to name a few.

Though the problem (2.10) is not in the form of (2.6), it can be transformed to (2.6) after introducing auxiliary variables and constraints. For example, introducing $y = x$, the problem (2.10) becomes

$$\begin{aligned} &\min_{x,y} \ f(Ax - b) + g(y) \\ &\text{s.t. } x - y = 0. \end{aligned} \tag{2.11}$$

Alternatively, one may also introduce $y = Ax - b$ to reformulate (2.10) as

$$\begin{aligned} &\min_{x,y} \ g(x) + f(y) \\ &\text{s.t. } Ax - y = b. \end{aligned} \tag{2.12}$$

Then ADMM is readily applicable to either (2.11) or (2.12). The way of reformulating the problems should be chosen wisely so that the resulting subproblems can be carried out efficiently.

2.3.2 Sparse and Low-Rank Optimization

In recent years, the problems of finding *sparse* or/and *low-rank* solutions have received tremendous attention from researchers and engineers, particularly those in the areas of compressive sensing, signal and image processing, machine learning and statistics. Many of these problems are posed as convex regularization models, where the ℓ_1-norm or *nuclear norm* are often used as the regularization function to enforce sparsity or low-rankness in the solutions. For example, we consider the following problems:

- **Sparse recovery** [11]:

$$\min_{x \in \mathbb{R}^n} \ \|x\|_1 + \frac{1}{2\mu}\|Ax - b\|_2^2, \tag{2.13}$$

where $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $\mu > 0$ is a regularization parameter. Here the ℓ_1 norm $\|x\|_1 := \sum_{i=1}^n |x_i|$ is known to promote sparse solutions. In compressive sensing, this problem is commonly used for reconstructing sparse signals from a relatively small number of (noisy) measurements. In machine learning and statistics, it is also known as the *Lasso* problem [56] for feature selection.

- **Low-rank matrix recovery/matrix completion** [7]:

$$\min_X \ \|X\|_* + \frac{1}{2\mu}\|\mathcal{A}(X) - b\|_2^2, \tag{2.14}$$

where $X \in \mathbb{R}^{m\times n}$ is an unknown low-rank matrix, $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is a linear operator, $b \in \mathbb{R}^p$ is our observed data, and $\mu > 0$ is a regularization parameter. Here, the nuclear norm $\|X\|_*$ denotes the sum of singular values of X which is well known for promoting low-rank solutions.

Such problems pose computational challenges due to the non-smoothness of the regularization functions as well as the often large scale of the problems. Traditional optimization methods, such as the interior-point methods, often cannot take advantage of the structures of the problems and are computationally expensive to handle very large problems.

We illustrate how ADMM can be effectively applied to these problems, giving rise to very efficient and scalable algorithms. Let us take the sparse recovery problem (2.13) as an illustrative example. By introducing the auxiliary variable $y = x$, (2.13) can be reformulated as:

$$\begin{aligned} &\min_{x,y} \ \|y\|_1 + \frac{1}{2\mu}\|Ax - b\|_2^2 \\ &\text{s.t. } x - y = 0. \end{aligned} \tag{2.15}$$

Applying ADMM to the above problem yields the following iterations:

$$y^{k+1} = \arg\min_{y} \|y\|_1 + \frac{\beta}{2}\left\|y - (x^k - \lambda^k/\beta)\right\|_2^2, \tag{2.16}$$

$$x^{k+1} = \arg\min_{x} \frac{1}{2\mu}\|Ax - b\|_2^2 + \frac{\beta}{2}\left\|x - y^{k+1} - \lambda^k/\beta\right\|_2^2, \tag{2.17}$$

$$\lambda^{k+1} = \lambda^k - \beta(x^{k+1} - y^{k+1}). \tag{2.18}$$

Note that the y-subproblem (2.16) admits a simple closed-form solution by the well-known *shrinkage* or *soft-thresholding* operator. For the low-rank matrix recovery problem (2.14), the application of ADMM follows exactly the same, except that the closed-form solution of the Y-subproblems takes a slightly different form by the so-called *singular value soft-thresholding*.

The x-subproblem (2.17) solves a quadratic problem that involves inverting a matrix $(\beta I + A^\top A/\mu)$ or solving a linear system. We may cache an initial matrix factorization to make the x-subproblems much cheaper to carry out. For very large problems where caching matrix factorization may not be affordable, it is more efficient to just solve the x-subproblem approximately, such as taking one gradient descent step as suggested in [61]. Though more iterations may be needed due to the less accurate subproblem, the entire algorithm runs faster because the subproblems can be carried out in much less time. This observation motivates us to generalize the ADMM to allow more options of solving the subproblems, possibly less exactly but faster, in order to make ADMM more efficient with wider applicability. We will discuss such generalization to ADMM and analyze its convergence in Chapter 3 and Chapter 4.

2.3.3 Consensus and Sharing Optimization

Consider in a network of N nodes, the problem of minimizing the sum of N functions, one from each node, over a common variable x. This problem can be written as

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{N} f_i(x). \tag{2.19}$$

Let each node i keep vector $x_i \in \mathbb{R}^n$ as its copy of x. To reach a consensus among x_i, $i = 1, \ldots, N$, a common approach is to introduce a global common variable y and get

$$\min_{\{x_i\},y} \sum_{i=1}^{N} f_i(x_i), \quad \text{s.t. } x_i - y = 0,\ i = 1, \ldots, N. \tag{2.20}$$

This is the well-known global consensus problem; see [4] for a review. With an objective function g on the global variable y, we have the global variable consensus problem with regularization:

$$\min_{\{x_i\},y} \sum_{i=1}^{N} f_i(x_i) + g(y), \quad \text{s.t. } x_i - y = 0,\ i = 1, \ldots, N, \tag{2.21}$$

where $g(y)$ is a convex function,

The following sharing problem is also nicely reviewed in [4]:

$$\min_{\{x_i\},y} \sum_{i=1}^{N} f_i(x_i) + g\left(\sum_{i=1}^{N} y_i\right), \quad \text{s.t. } x_i - y_i = 0,\ i = 1, \ldots, N, \tag{2.22}$$

where f_i's are local cost functions and g is the shared cost function by all the nodes i.

ADMM applied to the problems (2.20), (2.21) and (2.22) is particularly suitable for distributed implementation, since the x-subproblem can be decomposed into N independent x_i-subproblems, and the update to the multiplier λ can also be done at each node i. We will discuss this topic in more details in Chapter 5.

2.4 Notation and Preliminary

We let $\langle \cdot, \cdot \rangle$ denote the standard inner product in the Euclidean space. That is, for any $x = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^n$ and $y = (y_1, y_2, \ldots, y_n)^\top \in \mathbb{R}^n$,

$$\langle x, y \rangle := x^\top y = \sum_{i=1}^{n} x_i y_i. \tag{2.23}$$

We use $\| \cdot \|$ to denote the ℓ_2-norm. That is, for a vector $x = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^n$,

$$\|x\| := \sqrt{x^\top x} = \sqrt{\sum_{i=1}^{n} x_i^2}. \tag{2.24}$$

For a matrix $M \in \mathbb{R}^{m \times n}$, $\|M\|$ denotes the spectral norm, i.e., the largest singular value of M:

$$\|M\| := \sqrt{\lambda_{\max}(M^\top M)}. \tag{2.25}$$

We let $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the smallest and largest eigenvalues of a real symmetric matrix, respectively.

For a positive definite matrix $G \in \mathbb{R}^{n \times n}$, we define the *G-norm* as follows:

$$\|x\|_G := \sqrt{x^\top G x}, \ \forall x \in \mathbb{R}^n. \tag{2.26}$$

If the matrix G is positive semi-definite, then $\| \cdot \|_G$ is a semi-norm.

Let us consider an extended real-value function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, and let $\mathrm{dom} f$ denote its domain. We say the function is *proper* if there is at least one $x \in \mathrm{dom} f$ such that $f(x) < +\infty$. We say the function is *closed* if for any $\alpha \in \mathbb{R}$ the set $\{x \in \mathrm{dom} f : f(x) \leq \alpha\}$ is a closed set.

The function is *convex* if $\mathrm{dom} f$ is a convex set and the following inequality holds for all $t \in [0, 1]$ and all $x, y \in \mathrm{dom} f$:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y). \tag{2.27}$$

The function is called *strictly convex* if the above inequality holds strictly (with "<") for all $t \in (0,1)$ and all $x \neq y$ such that $f(x) < +\infty$ and $f(y) < +\infty$. The function is called *strongly convex* with constant $\nu > 0$ if the function $f(x) - \frac{\nu}{2}\|x\|^2$ is convex, or equivalently, if the following inequality holds for all $t \in [0,1]$ and all $x, y \in \mathrm{dom} f$:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) - \frac{1}{2}\nu t(1-t)\|x-y\|^2. \tag{2.28}$$

For a convex function f, we let the *subdifferential* (i.e., the set of all subgradients) of f at $x \in \mathrm{dom} f$ be denoted by $\partial f(x)$:

$$\partial f(x) := \left\{ s \in \mathbb{R}^n : \ s^\top (y-x) \leq f(y) - f(x), \ \forall y \in \mathrm{dom} f \right\}. \tag{2.29}$$

If f is strongly convex with constant $\nu > 0$, then it can be shown that for any $x \in \mathrm{dom} f$ and $s \in \partial f(x)$, the following holds:

$$f(y) - f(x) \geq s^\top (y-x) + \frac{\nu}{2}\|x-y\|^2. \tag{2.30}$$

For a differentiable function f, the gradient ∇f is called *Lipschitz continuous* with constant $L > 0$ if

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x-y\|, \ \forall x, y \in \mathrm{dom} f. \tag{2.31}$$

Let us review some basic properties of convex functions which will be used frequently in the later chapters.

Lemma 2.1 (monotonicity of subdifferential)**.** *If function f is convex, then for any $x, y \in \mathrm{dom}\ f$, we have*

$$\langle x-y, s-t \rangle \geq 0, \ \forall s \in \partial f(x), \ t \in \partial f(y). \tag{2.32}$$

If function f is strongly convex with constant $\nu > 0$, then for any $x, y \in \mathrm{dom}\ f$, we have

$$\langle x-y, s-t \rangle \geq \nu\|x-y\|^2, \ \forall s \in \partial f(x), \ t \in \partial f(y). \tag{2.33}$$

Proof. For brevity, we only prove (2.33) since it reduces to (2.32) when $\nu = 0$. By (2.30), for any $s \in \partial f(x)$ and $t \in \partial f(y)$, we have

$$f(y) - f(x) - s^\top (y - x) \geq \frac{\nu}{2}\|x - y\|^2,$$
$$f(x) - f(y) - t^\top (x - y) \geq \frac{\nu}{2}\|x - y\|^2.$$

Adding these two inequalities together yields (2.33). □

Lemma 2.2. *If a convex function f has Lipschitz continuous gradient ∇f with constant $L > 0$, then the following inequality holds:*

$$\langle x - y,\ \nabla f(x) - \nabla f(y)\rangle \geq \frac{1}{L}\|\nabla f(x) - \nabla f(y)\|^2,\ \forall x, y \in \mathrm{dom} f. \tag{2.34}$$

For the sake of brevity, we omit the proof which can be found in many convex analysis books, for example [44].

Chapter 3

Generalized ADMM with Simplified Subproblems

In this chapter, we consider a generalized ADMM framework that allows various options of simplifying the subproblems. The generalization of ADMM is of great practical importance because it brings more flexibility and efficiency in carrying out the iterations, thereby making the entire algorithm run faster and scale better for large problems.

First, in Section 3.1, we discuss the motivation of the generalization and give some motivating examples. In Section 3.2, we present the technical details of the generalized ADMM framework, as well as comparison to some closely related work [30, 62]. Then, in Section 3.3, we establish the global convergence of the generalized ADMM under some standard assumptions. In Section 3.4, we briefly review some existing results on the convergence rates of ADMM. Furthermore, in Section 3.5, we introduce a simple technique to improve an existing $O(1/k)$ convergence rate to $o(1/k)$, where k is the number of iterations.

3.1 Motivation

The efficiency of ADMM often relies on the fact that the original problem can be decomposed into easier subproblems which admit efficient solvers or simple closed-form solutions. In many applications, however, the subproblems of ADMM are not always easy to carry out. Especially for many large-scale problems, solving the subproblems

exactly may become computationally expensive, and applying ADMM straightforwardly may be inefficient. If one spends too much time on solving each single subproblem, the entire algorithm will take a long time to finish. Often, even though some subproblems are expensive to solve *exactly*, it is much cheaper to compute *approximate* solutions to the subproblems which are still good enough to guarantee the convergence. Although it may take more iterations to converge due to less accurate subproblems, the entire algorithm can finish in less time since each iteration runs much faster.

In practice, many variants of ADMM have been developed in which the subproblems are solved inexactly but faster in different ways. Below we give several useful examples on how to come up with efficient methods to solve the subproblems approximately. For the sake of brevity, we consider only the x-subproblem:

$$\min_x \ f(x) - (\lambda^k)^T Ax + \frac{\beta}{2}\|Ax + By^{k+1} - b\|^2, \tag{3.1}$$

while the same arguments will apply to the y-subproblem as well.

- **Prox-linear ADMM.** The Prox-linear method was introduced in [10], which linearizes the quadratic term $\frac{\beta}{2}\|Ax + By^{k+1} - b\|^2$ in (3.1) at $x = x^k$ and adds a proximal term $\|x - x^k\|^2$. That is, it solves the following problem:

$$\min_x \ f(x) + (p^k)^T(x - x^k) + \frac{\beta}{2\tau}\|x - x^k\|^2, \tag{3.2}$$

  where $\tau > 0$ is a proximal parameter and $p^k := \beta A^T(Ax^k + By^{k+1} - b - \lambda^k/\beta)$ is the gradient of the last two terms of (3.1) at $x = x^k$. Compared to the original x-subproblem (3.1), it essentially replaces the Hessian matrix $\beta A^T A$ of the quadratic term by an identity matrix $\frac{\beta}{\tau}\mathbf{I}$.

  Prox-linear subproblems are much easier to compute in various applications.

For example, if f is a *separable*[*] function, problem (3.2) reduces to a set of independent one-dimensional problems. In particular, if f is ℓ_1 norm, the solution is given in the closed form by the so-called soft-thresholding. If f is the matrix nuclear norm, then the solution is given by singular-value soft-thresholding. If $f(x) = \|\Phi x\|_1$ where Φ is an orthogonal operator or a tight frame, (3.2) also has a closed-form solution. If f is total variation, (3.2) can be solved by graph-cut [5, 26]. There are a large number of such examples in signal processing, imaging, statistics, machine learning, etc.

- **Gradient-descent ADMM.** In many situations, the function f is *quadratic* and hence the x-subproblem amounts to solving a linear system. When one must solve a large, nontrivial linear system at every iteration, the overall computational cost can be very expensive. Alternatively, one may simply take a gradient descent step to compute an approximate solution:

$$x^{k+1} = x^k - \alpha g^k, \tag{3.3}$$

where $\alpha > 0$ is a step size and

$$g^k := \nabla f(x^k) + \beta A^T(Ax^k + By^{k+1} - b - \lambda^k/\beta) \tag{3.4}$$

is the gradient of the augmented Lagrangian $\mathcal{L}_\mathcal{A}(x, y^{k+1}, \lambda^k)$ at $x = x^k$. Apparently, taking one gradient step only requires several matrix-vector multiplications, which significantly reduces the cost and has a clear speed advantage.

- **ADMM with fast approximation of A (and/or B).** The x-subproblem contains the quadratic term $\frac{\beta}{2}x^T A^T Ax$. Sometimes, replacing $A^T A$ by a certain

[*]We call a function $f : \mathbb{R}^n \to \mathbb{R}$ *separable* if $f(x) = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n)$ for $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, where $f_i : \mathbb{R} \to \mathbb{R}$, $\forall i = 1, 2, \ldots, n$

symmetric matrix $D \approx A^T A$ makes the problem easier to compute. For example, this approach is useful when $A^T A$ is nearly diagonal (D is the diagonal matrix), or is an orthogonal matrix plus error (D is the orthogonal matrix), as well as when an off-the-grid operator A can be approximated by its on-the-grid counterpart that has very fast implementations (e.g., the discrete Fourier transforms and FFT).

3.2 Generalized ADMM

As discussed above, variants of ADMM that allow subproblems to be solved approximately and faster are very important to the applications in which it is expensive to exactly solve either the x-subproblem or the y-subproblem, or both of them. For this reason, we present a generalized ADMM framework (Algorithm 2 below) by allowing various ways of simplifying the subproblems. In particular, it includes not only the classic ADMM (Algorithm 1), but also the aforementioned *Prox-linear ADMM*, *Gradient-descent ADMM*, *ADMM with fast approximation of A and/or B* as well as combinations of them as special cases.

Algorithm 2: Generalized ADMM

Choose $Q \succeq 0$ and a symmetric matrix P (possibly indefinite);

Initialize $x^0,\ y^0,\ \lambda^0,\ \beta > 0, \gamma > 0$;

for $k = 0,\ 1, \ldots$ **do**

$y^{k+1} = \arg\min_y \mathcal{L}_{\mathcal{A}}(x^k, y, \lambda^k) + \frac{1}{2}(y - y^k)^T Q(y - y^k)$;

$x^{k+1} = \arg\min_x \mathcal{L}_{\mathcal{A}}(x, y^{k+1}, \lambda^k) + \frac{1}{2}(x - x^k)^T P(x - x^k)$;

$\lambda^{k+1} = \lambda^k - \gamma\beta(Ax^{k+1} + By^{k+1} - b)$.

Compared to Algorithm 1, Algorithm 2 adds $\frac{1}{2}\|y - y^k\|_Q^2$ and $\frac{1}{2}\|x - x^k\|_P^2$ to the

y- and x-subproblems, respectively, and assigns γ as the step size for the update of λ. Here, we slightly abuse the notion $\|x\|_M^2 := x^T M x$ as we allow any *symmetric matrix* M. Different choices of P and Q are overviewed in the next subsection. They can make steps 4 and 5 of Algorithm 2 easier than those of Algorithm 1. Obviously, Algoirthm 2 reduces to Algorithm 1 when $P = \mathbf{0}$, $Q = \mathbf{0}$ and $\gamma = 1$.

We do not fix $\gamma = 1$ like in most of the ADMM literature since γ plays an important role in convergence and speed. For example, when $P = \mathbf{0}$ and $Q = \mathbf{0}$, any $\gamma \in (0, (\sqrt{5}+1)/2)$ guarantees the convergence of Algorithm 2 [22], but $\gamma = 1.618 \approx (\sqrt{5}+1)/2$ makes it converge noticeably faster than $\gamma = 1$. As we will show in Section 3.3, the range of γ depends on P and Q, as well as β. When P is indefinite, γ must be smaller than 1 or the iteration may diverge.

3.2.1 Choices of P and Q.

The general goal is to wisely choose P and Q so that the subproblems of Algorithm 2 become much easier to carry out and the entire algorithm runs in less time. Let us give a few examples of matrix P in step 5 of Algorithm 2. These examples also apply to Q in step 4 as well. Note that P and Q can be different.

- **Exact ADMM.** If the original x-subproblem in Algorithm 1 is already easy to compute exactly, then one can simply set $P = \mathbf{0}$.

- **Prox-linear ADMM.** Setting

$$P = \frac{\beta}{\tau} I - \beta A^T A, \tag{3.5}$$

  gives rise to the prox-linear problem (3.2).

- **Gradient-descent ADMM.** When function f is *quadratic*, letting

$$P = \frac{1}{\alpha} I - H_f - \beta A^T A, \text{ where } H_f := \nabla^2 f(x) \succeq 0, \tag{3.6}$$

yields the following x-subproblem:

$$\min_x \ (g^k)^T (x - x^k) + \frac{1}{2\alpha} \|x - x^k\|_2^2, \tag{3.7}$$

where g^k is the gradient given by (3.4). Clearly, it amounts to the gradient descent step (3.3).

- **ADMM with fast approximation of A (and/or B).** To replace $A^T A$ by a certain symmetric matrix $D \approx A^T A$, one can let

$$P = \beta(D - A^T A). \tag{3.8}$$

This choice of P effectively turns $\frac{\beta}{2} x^T A^T A x$ into $\frac{\beta}{2} x^T D x$ since

$$\begin{aligned} &\frac{\beta}{2} \|Ax + By^{k+1} - b\|_2^2 + \frac{1}{2} \|x - x^k\|_P^2 \\ =&\frac{\beta}{2} x^T D x + [\text{terms linear in } x] + [\text{terms independent of } x]. \end{aligned}$$

Note that P can be indefinite. This approach also applies to step 4 of Algorithm 2 as long as $Q \succ 0$.

In ADMM, the two subproblems can be solved in either order (but fixed throughout the iterations). However, when one subproblem is solved less exactly than the other, Algorithm 2 tends to run faster if the *less* exact one is solved *later* — assigned as step 5 of Algorithm 2 — because at each iteration, the ADMM updates the variables in the Gauss-Seidel fashion. If the less exact one runs first, its relatively inaccurate solution will then affect the more exact step, making its solution also inaccurate. Since the less exact subproblem should be assigned as the later step 5, more choices of P are needed than Q, which is the case in Algorithm 2.

3.2.2 Related Work

Let us overview two works very related to Algorithm 2. Both works [30, 62] consider adding proximal terms to the subproblems, where P and Q are restricted to positive semi-definite matrices. In [30], the convergence analysis is restricted to the case of $\gamma = 1$ and *differentiable* functions f and g; on the other hand, the quadratic penalty term of augmented Lagrangian is further generalized to $\|Ax+By-b\|^2_{H_k}$ for a sequence of bounded positive definite matrices $\{H_k\}$. The work [62] extends the scalar γ to a positive definite matrix C and establishes convergence assuming that $A = I$ and the smallest eigenvalue of C is no greater than 1, which corresponds to $\gamma \leq 1$ when $C = \gamma I$.

Our work makes meaningful extensions to the existing convergence theory in [30, 62]. Specifically, the step size γ is less restrictive, and P is allowed to be indefinite. These extensions translate to faster convergence and more options of solving the x-subproblem efficiently. While there is no rate of convergence given in [30, 62], we will further establish linear convergence rates in Chapter 4.

3.3 Global Convergence

In this section, we show the global convergence of Algorithm 2. The proof steps are similar to the existing convergence theory in [30, 62] but are adapted to Algorithm 2 with several extensions. The analysis in this section will also be used frequently in the next chapter to establish linear convergence under additional assumptions.

3.3.1 Assumptions

Throughout the rest of this chapter as well as Chapter 4, we make the following standard assumptions.

Assumption 3.1. *There exists a saddle point* $u^* := (x^*, y^*, \lambda^*)$ *to problem* (2.6)*, namely,* x^**,* y^**, and* λ^* *satisfy the KKT conditions:*

$$A^T\lambda^* \in \partial f(x^*), \tag{3.9}$$

$$B^T\lambda^* \in \partial g(y^*), \tag{3.10}$$

$$Ax^* + By^* - b = 0. \tag{3.11}$$

These conditions can be written in a more compact form by the following variational inequality:

$$f(x) + g(y) - f(x^*) - g(y^*) + (u - u^*)^\top H(u^*) \geq 0, \ \forall u, \tag{3.12}$$

where

$$H(u) := \begin{pmatrix} -A^\top\lambda \\ -B^\top\lambda \\ Ax + By - b \end{pmatrix}.$$

When assumption 3.1 fails to hold, the ADMM method has either unsolvable or unbounded subproblems or a diverging sequence of λ^k.

Assumption 3.2. *Functions* f *and* g *are closed, proper and convex.*

We define scalars ν_f and ν_g as the modulus of f and g, respectively. Following from Lemma 2.1, they satisfy

$$\langle s_1 - s_2, x_1 - x_2\rangle \geq \nu_f\|x_1 - x_2\|^2, \quad \forall x_1,\ x_2,\ s_1 \in \partial f(x_1),\ s_2 \in \partial f(x_2), \tag{3.13}$$

$$\langle t_1 - t_2, y_1 - y_2\rangle \geq \nu_g\|y_1 - y_2\|^2, \quad \forall y_1,\ y_2,\ \ t_1 \in \partial g(y_1),\ \ t_2 \in \partial g(y_2). \tag{3.14}$$

From the convexity of f and g, it follows that $\nu_f \geq 0$ and $\nu_g \geq 0$. They are (strictly) positive if the functions are *strongly* convex.

Throughout this chapter, since f and g are allowed to be any convex functions, we can treat $\nu_f = 0$ and $\nu_g = 0$ for the worst case. In Chapter 4, we will further assume $\nu_f > 0$ or/and $\nu_g > 0$, as strong convexity of at least one of the functions is needed to show *linear* convergence.

3.3.2 Convergence Analysis

For notation simplicity, we introduce

$$\hat{\lambda} := \lambda^k - \beta(Ax^{k+1} + By^{k+1} - b). \tag{3.15}$$

If $\gamma = 1$, then $\hat{\lambda} = \lambda^{k+1}$; otherwise,

$$\hat{\lambda} - \lambda^{k+1} = (\gamma - 1)\beta(Ax^{k+1} + By^{k+1} - b) = (1 - \frac{1}{\lambda})(\lambda^k - \lambda^{k+1}). \tag{3.16}$$

This relation between $\hat{\lambda}$ and λ^{k+1} is used frequently in our analysis. Let

$$u^* := \begin{pmatrix} x^* \\ y^* \\ \lambda^* \end{pmatrix}, \; u^k := \begin{pmatrix} x^k \\ y^k \\ \lambda^k \end{pmatrix}, \; \hat{u} := \begin{pmatrix} x^{k+1} \\ y^{k+1} \\ \hat{\lambda} \end{pmatrix}, \; \text{for } k = 0, 1, \ldots, \tag{3.17}$$

where u^* is a KKT point, u^k is the current point, and $\hat{u}$ is the next point *as if* $\gamma = 1$, and

$$G_0 := \begin{pmatrix} I_n & & \\ & I_m & \\ & & \gamma I_p \end{pmatrix}, \; G_1 := \begin{pmatrix} \hat{P} & & \\ & Q & \\ & & \frac{1}{\beta} I_p \end{pmatrix}, \; G := G_0^{-1} G_1 = \begin{pmatrix} \hat{P} & & \\ & Q & \\ & & \frac{1}{\beta\gamma} I_p \end{pmatrix}, \tag{3.18}$$

where we recall $\hat{P} = P + \beta A^T A$. From these definitions it follows

$$u^{k+1} = u^k - G_0(u^k - \hat{u}). \tag{3.19}$$

We choose P, Q and β such that $\hat{P} \succeq 0$ and $Q \succeq 0$. Hence $G \succeq 0$ and $\|\cdot\|_G$ is a (semi-)norm. The definitions of the matrix G and the G-norm are similar to those in the work [35]. Our analysis is based on bounding the error $\|u^k - u^*\|_G^2$ and estimate its decrease.

Lemma 3.1. *Under Assumptions 3.1 and 3.2, the sequence $\{u^k\}$ of algorithm 2 obeys*

i)

$$A^T\hat{\lambda} + P(x^k - x^{k+1}) \in \partial f(x^{k+1}), \tag{3.20}$$

$$B^T\left(\hat{\lambda} - \beta A(x^k - x^{k+1})\right) + Q(y^k - y^{k+1}) \in \partial g(y^{k+1}). \tag{3.21}$$

ii)

$$\langle x^{k+1} - x^*,\ A^T(\hat{\lambda} - \lambda^*) + P(x^k - x^{k+1})\rangle \geq \nu_f\|x^{k+1} - x^*\|^2, \tag{3.22}$$

$$\langle y^{k+1} - y^*,\ B^T\left(\hat{\lambda} - \lambda^* - \beta A(x^k - x^{k+1})\right) + Q(y^k - y^{k+1})\rangle \geq \nu_g\|y^{k+1} - y^*\|^2. \tag{3.23}$$

iii)

$$A(x^{k+1} - x^*) + B(y^{k+1} - y^*) = \frac{1}{\beta}(\lambda^k - \hat{\lambda}). \tag{3.24}$$

iv)

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq h(u^k - \hat{u}) + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2, \tag{3.25}$$

where

$$\begin{aligned} h(u^k - \hat{u}) :=& \|x^k - x^{k+1}\|_{\hat{P}}^2 + \|y^k - y^{k+1}\|_Q^2 + \frac{2-\gamma}{\beta}\|\lambda^k - \hat{\lambda}\|^2 \\ &+ 2(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1}). \end{aligned} \tag{3.26}$$

Proof. i) By the optimality conditions for the subproblems of Algorithm 2 and using (3.15), we obtain (3.20) and (3.21) immediately.

ii) By the convexity of f (3.13), combining the optimality conditions (3.9) and (3.20) yields (3.22). Similarly, by the convexity of g (3.14), combing the optimality conditions (3.10) and (3.21) yields (3.23).

iii) The equation (3.24) follows directly from (3.11) and (3.15).

iv) By adding (3.22) and (3.23) and using (3.24), we have

$$\begin{aligned}&\frac{1}{\beta}\langle\lambda^k-\hat{\lambda},\ \hat{\lambda}-\lambda^*-\beta A(x^k-x^{k+1})\rangle+\langle x^{k+1}-x^*,\ \hat{P}(x^k-x^{k+1})\rangle\\&+\langle y^{k+1}-y^*,\ Q(y^k-y^{k+1})\rangle\geq\nu_f\|x^{k+1}-x^*\|^2+\nu_g\|y^{k+1}-y^*\|^2,\end{aligned}\tag{3.27}$$

which can be simplified as

$$\begin{aligned}&(\hat{u}-u^*)^TG_1(u^k-\hat{u})\\\geq&\langle A(x^k-x^{k+1}),\ \lambda^k-\hat{\lambda}\rangle+\nu_f\|x^{k+1}-x^*\|^2+\nu_g\|y^{k+1}-y^*\|^2.\end{aligned}\tag{3.28}$$

By rearranging the terms, we have

$$\begin{aligned}(u^k-u^*)^TG_1(u^k-\hat{u})\geq\|u^k-\hat{u}\|^2_{G_1}&+\langle A(x^k-x^{k+1}),\lambda^k-\hat{\lambda}\rangle\\&+\nu_f\|x^{k+1}-x^*\|^2+\nu_g\|y^{k+1}-y^*\|^2.\end{aligned}\tag{3.29}$$

Then, from (3.19) and the identity

$$\|a-c\|^2_G-\|b-c\|^2_G=2(a-c)^TG(a-b)-\|a-b\|^2_G,\tag{3.30}$$

it follows that

$$\|u^k-u^*\|^2_G-\|u^{k+1}-u^*\|^2_G=2(u^k-u^*)^TG_1(u^k-\hat{u})-\|G_0(u^k-\hat{u})\|^2_G.\tag{3.31}$$

Substituting (3.29) into the right-hand side of the above equation, we have

$$\begin{aligned}\|u^k-u^*\|^2_G-\|u^{k+1}-u^*\|^2_G\geq\ &2\|u^k-\hat{u}\|^2_{G_1}-\|u^k-\hat{u}\|^2_{G_1G_0}\\&+2\langle A(x^k-x^{k+1}),\lambda^k-\hat{\lambda}\rangle+2\nu_f\|x^{k+1}-x^*\|^2+2\nu_g\|y^{k+1}-y^*\|^2,\end{aligned}\tag{3.32}$$

and thus (3.25) follows immediately.

□

In the next theorem, we show that $\|u^k - u^*\|_G^2$ has sufficient descent. Technically, it is done through bounding $h(u^k - \hat{u})$ from zero by applying the Cauchy-Schwarz inequality to its cross term $2(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1})$. If $P = \mathbf{0}$, a more refined bound is obtained to give γ a wider range of convergence.

Theorem 3.1 (Sufficient descent of $\|u^k - u^*\|_G^2$)**.** *Assume Assumptions 3.1 and 3.2.*

i) When $P \neq \mathbf{0}$, if γ obeys

$$(2-\gamma)P \succ (\gamma - 1)\beta A^T A \tag{3.33}$$

(see Remark 3.1 below for simplification), then there exists $\eta > 0$ such that

$$\begin{aligned} &\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \\ \geq &\eta\|u^k - u^{k+1}\|_G^2 + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2. \end{aligned} \tag{3.34}$$

ii) When $P = \mathbf{0}$, if

$$\gamma \in \left(0, \frac{1+\sqrt{5}}{2}\right), \tag{3.35}$$

then there exist $\eta > 0$ such that

$$\begin{aligned} &\left(\|u^k - u^*\|_G^2 + \frac{\beta}{\rho}\|r^k\|^2\right) - \left(\|u^{k+1} - u^*\|_G^2 + \frac{\beta}{\rho}\|r^{k+1}\|^2\right) \\ \geq\ &\eta\|u^k - u^{k+1}\|_G^2 + 2\nu_f\|x^k - x^{k+1}\|^2 + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2, \end{aligned} \tag{3.36}$$

where r^k is the residual at iteration k:

$$r^k := Ax^k + By^k - b.$$

If we set $\gamma = 1$, then we have

$$\begin{aligned}\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq & \|u^k - u^{k+1}\|_G^2 + 2\nu_f\|x^k - x^{k+1}\|^2 \\ & + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2.\end{aligned} \tag{3.37}$$

Proof. i) By the Cauchy-Schwarz inequality, we have

$$2(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1}) \geq -\frac{1}{\rho}\|A(x^k - x^{k+1})\|^2 - \rho\|\lambda^k - \hat{\lambda}\|^2, \ \forall \rho > 0. \tag{3.38}$$

Substituting (3.38) into (3.25) and using $\frac{1}{\gamma}(\lambda^k - \lambda^{k+1}) = \lambda^k - \hat{\lambda}$, we have

$$\begin{aligned}&\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \\ \geq\ & \|x^k - x^{k+1}\|^2_{\hat{P} - \frac{1}{\rho}A^TA} + \|y^k - y^{k+1}\|_Q^2 + \left(\frac{2-\gamma}{\beta} - \rho\right)\frac{1}{\gamma^2}\|\lambda^k - \lambda^{k+1}\|^2 \\ & + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2, \ \forall \rho > 0.\end{aligned} \tag{3.39}$$

To show that (3.34) holds for a certain $\eta > 0$, we only need $\hat{P} - \frac{1}{\rho}A^TA \succ 0$ and $\frac{2-\gamma}{\beta} - \rho > 0$ for a certain $\rho > 0$, which is true if and only if we have $\hat{P} \succ \frac{\beta}{2-\gamma}A^TA$ or, equivalently, (3.33).

ii) For $P = \mathbf{0}$, we first derive a lower bound for the cross term $(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1})$. Applying (3.20) at two consecutive iterations with $P = \mathbf{0}$ and in light of the definition of $\hat{\lambda}$, we have

$$\begin{cases} A^T[\lambda^{k-1} - \beta(Ax^k + By^k - b)] \in \partial f(x^k), \\ A^T\hat{\lambda} \in \partial f(x^{k+1}). \end{cases} \tag{3.40}$$

The difference of the two terms on the left in (3.40) is

$$A^T[\lambda^{k-1} - \beta(Ax^k + By^k - b) - \hat{\lambda}] = A^T(\lambda^k - \hat{\lambda}) - (1-\gamma)\beta A^T(Ax^k + By^k - b). \tag{3.41}$$

By (3.40), (3.41) and (3.13), we get

$$\langle A^T(\lambda^k - \hat{\lambda}), x^k - x^{k+1}\rangle - \langle(1-\gamma)\beta A^T(Ax^k + By^k - b), x^k - x^{k+1}\rangle \geq \nu_f\|x^k - x^{k+1}\|^2, \tag{3.42}$$

to which applying the Cauchy-Schwarz inequality gives

$$\begin{aligned}
&(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1}) \\
\geq &\langle \sqrt{\beta}(Ax^k + By^k - b), (1-\gamma)\sqrt{\beta}A(x^k - x^{k+1})\rangle + \nu_f \|x^k - x^{k+1}\|^2 \\
\geq &-\frac{\beta}{2\rho}\|Ax^k + By^k - b\|^2 - \frac{(1-\gamma)^2\beta\rho}{2}\|A(x^k - x^{k+1})\|^2 + \nu_f\|x^k - x^{k+1}\|^2,
\end{aligned} \tag{3.43}$$

for any $\rho > 0$. Substituting (3.43) into (3.25) and using $\hat{P} = P + \beta A^T A = \beta A^T A$ and the definition of $\hat{\lambda}$, we have

$$\begin{aligned}
&\|u^k - u^*\|_G^2 + \frac{\beta}{\rho}\|Ax^k + By^k - b\|^2 \\
\geq\ &\|u^{k+1} - u^*\|_G^2 + \frac{\beta}{\rho}\|Ax^{k+1} + By^{k+1} - b\|^2 \\
&+ \beta\left(2 - \gamma - \frac{1}{\rho}\right)\|Ax^{k+1} + By^{k+1} - b\|^2 + \beta\left(1 - (1-\gamma)^2\rho\right)\|A(x^k - x^{k+1})\|^2 \\
&+ \|y^k - y^{k+1}\|_Q^2 + 2\nu_f\|x^k - x^{k+1}\|^2 + 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2.
\end{aligned} \tag{3.44}$$

To prove such $\eta > 0$ exists for (3.36), we only need the existence of $\rho > 0$ such that $2 - \gamma - \frac{1}{\rho} > 0$ and $1 - (1-\gamma)^2\rho > 0$, which holds if and only if $2 - \gamma > (1-\gamma)^2$ or, equivalently, $\gamma \in (0, \frac{1+\sqrt{5}}{2})$.

In this case of $P = \mathbf{0}$, if we set $\gamma = 1$, (3.43) reduces to $(\lambda^k - \hat{\lambda})^T A(x^k - x^{k+1}) \geq \nu_f\|x^k - x^{k+1}\|^2$, which substituting into (3.25) gives (3.37) with $\eta = 1$.

□

Now the sufficient descent of $\|u^k - u^*\|_G^2$ in Theorem 3.1 are used to yield the global convergence of Algorithm 2.

Theorem 3.2 (Global convergence of Algorithm 2)**.** *Assume Assumptions 3.1 and 3.2, and that $\{u^k\}$ of Algorithm 2 is bounded (see Remark 3.2 below). For any γ*

satisfying its conditions given in Theorem 3.1, $\{u^k\}$ converges to a KKT point u^* of (2.6) in the G-norm, namely,

$$\|u^k - u^*\|_G \to 0.$$

It further follows that

(a) $\lambda^k \to \lambda^*$, regardless of the choice of P and Q;

(b) when $P \neq \mathbf{0}$, $x^k \to x^*$; otherwise, $Ax^k \to Ax^*$;

(c) when $Q \succ \mathbf{0}$, $y^k \to y^*$; when $Q = \mathbf{0}$, $By^k \to By^*$.

Proof. Being bounded, $\{u^k\}$ has a converging subsequence $\{u^{k_j}\}$. Let $\bar{u} = \lim_{j\to\infty} u^{k_j}$. Next, we will show $\bar{u}$ is a KKT point. Let u^* denote an arbitrary KKT point.

Consider $P \neq \mathbf{0}$ first. From (3.34) we conclude that $\|u^k - u^*\|_G^2$ is monotonically nonincreasing and thus converging, and due to $\eta > 0$, $\|u^k - u^{k+1}\|_G^2 \to 0$. In light of (3.18) where $\hat{P} \succeq 0$ and $Q \succeq 0$, we obtain $\lambda^k - \lambda^{k+1} \to 0$ or equivalently,

$$r^k = (Ax^{k+1} + By^{k+1} - b) \to 0, \quad \text{as } k \to \infty. \tag{3.45}$$

Now consider $P = \mathbf{0}$. From (3.36) we conclude that $\|u^k - u^*\|_G^2 + \frac{\beta}{\rho}\|r^k\|^2$ is monotonically nonincreasing and thus converging. Due to $\eta > 0$, $\|u^k - u^{k+1}\|_G^2 \to 0$, so $\lambda^k - \lambda^{k+1} \to 0$ and (3.45) holds as well. Consequently, $\|u^k - u^*\|_G^2$ also converges.

Therefore, by passing limit on (3.45) over the subsequence, we have for $P = 0$ or not:

$$A\bar{x} + B\bar{y} - b = 0. \tag{3.46}$$

Recall the optimality conditions (3.21) and (3.20):

$$B^T\hat{\lambda} - \beta B^T A(x^k - x^{k+1}) + Q(y^k - y^{k+1}) \in \partial g(y^{k+1}),$$
$$A^T\hat{\lambda} + P(x^k - x^{k+1}) \in \partial f(x^{k+1}).$$

Since $\|u^k - u^{k+1}\|_G^2 \to 0$, in light of the definition of G (3.18), we have the following:

- when $P = \mathbf{0}$, $A(x^k - x^{k+1}) \to \mathbf{0}$;
- when $P \neq \mathbf{0}$, the condition (3.33) guarantees $\hat{P} \succ \mathbf{0}$ and thus $x^k - x^{k+1} \to 0$;
- since $Q \succeq \mathbf{0}$, we obtain $Q(y^k - y^{k+1}) \to 0$.

In summary, $\beta B^T A(x^k - x^{k+1})$, $Q(y^k - y^{k+1})$, and $P(x^k - x^{k+1})$ are either 0 or converging to 0 in k, no matter $P = \mathbf{0}$ or not.

Now on both sides of (3.21) and (3.20) taking limit over the *subsequence* and applying Theorem 24.4 of [52], we obtain:

$$B^T \bar{\lambda} \in \partial g(\bar{y}), \tag{3.47}$$

$$A^T \bar{\lambda} \in \partial f(\bar{x}). \tag{3.48}$$

Therefore, together with (3.46), $\bar{u}$ satisfies the KKT condition of (2.6).

Since $\bar{u}$ is a KKT point, we can now let $u^* = \bar{u}$. From $u^{k_j} \to \bar{u}$ in j and the convergence of $\|u^k - u^*\|_G^2$ it follows $\|u^k - u^*\|_G^2 \to 0$ in k.

By the definition of G, $\|u^k - u^*\|_G^2 \to 0$ implies the following:

(a) $\lambda^k \to \lambda^*$, regardless of the choice of P and Q;

(b) when $P \neq \mathbf{0}$, condition (3.33) guarantees $\hat{P} \succ \mathbf{0}$ and thus $x^k \to x^*$; when $P = \mathbf{0}$, $Ax^k \to Ax^*$;

(c) when $Q \succ \mathbf{0}$, $y^k \to y^*$; when $Q = \mathbf{0}$, $By^k \to By^*$ following from (3.45) and (3.46).

□

Remark 3.1. *Let us discuss the conditions on γ. If $P \succ 0$, the condition (3.33) is always be satisfied for $0 < \gamma \leq 1$. However, in this case, γ can go greater than 1,*

which often leads to faster convergence in practice. If $P \not\succ 0$, the condition (3.33) requires γ to lie in $(0, \bar{\gamma})$ where $0 < \bar{\gamma} < 1$ depends on β, P, and A^TA. A larger β would allow a larger $\bar{\gamma}$.

In particular, in Prox-liner ADM where the x-subproblem is solved by (3.2), condition (3.33) is guaranteed by

$$\tau\|A\|^2 + \gamma < 2. \tag{3.49}$$

In Gradient-descent ADM where the x-subproblem has the form of (3.7), a sufficient condition for (3.33) is given by

$$\frac{\beta\|A\|^2}{\frac{1}{\alpha} - \|H_f\|} + \gamma < 2. \tag{3.50}$$

Remark 3.2. *The assumption on the boundedness of the sequence $\{u^k\}$ can be guaranteed by various conditions. Since (3.34) and (3.36) imply that $\|u^k - u^*\|_G^2$ is bounded, $\{u^k\}$ must be bounded if $\hat{P} \succ 0$ and $Q \succ 0$. Furthermore, if $P = \mathbf{0}$ and $Q = \mathbf{0}$, we have the boundedness of $\{(Ax^k, \lambda^k)\}$ (since $\|u^k - u^*\|_G^2$ is bounded) and that of $\{By^k\}$ by (3.24), so in this case, $\{u^k\}$ is bounded if*

(i) matrix A has full column rank whenever $P = \mathbf{0}$; and

(ii) matrix B has full column rank whenever $Q = \mathbf{0}$.

In addition, the boundedness of $\{u^k\}$ is guaranteed if the objective functions are coercive.

3.4 Existing Rate-of-Convergence Results

In this section, we review some existing results on the convergence rate of ADMM. Although there is extensive literature on ADMM and its applications, there are very few results on its rate of convergence until the very recent past. Work [24] shows

that for a Jacobi version of the ADMM applied to smooth functions with Lipschitz continuous gradients, the objective value descends at the rate of $O(1/k)$ and that of an accelerated version descends at $O(1/k^2)$. Then, work [25] establishes the same rates on a Gauss-Seidel version and requires only one of the two objective functions to be smooth with Lipschitz continuous gradient. But these two works only consider the model with the linear constraint coefficient matrices A and B being identity matrix or negative identity matrix.

Lately, work [36] shows that the optimality conditions of the ADMM based on a variational inequality converges at an $O(1/k)$ rate in an ergodic sense. Work [35] shows that $\|u^k - u^{k+1}\|^2$, where $u^k := (x^k, y^k, \lambda^k)$, of the ADMM converges at $O(1/k)$. Work [28] proves that the dual objective value of an accelerated version of ADMM descends at $O(1/k^2)$ under the assumption that the objective functions are strongly convex (one of them being quadratic).

Besides these sublinear rates, several linear rates have also been established. Work [16] shows that the ADMM applied to linear programming converges at a global linear rate. For quadratic programming, work [3] presents an analysis leading to a conjecture that the ADMM should converge linearly near the optimal solution.

Work [38] proves the linear convergence of ADMM in a different approach. The linear convergence in [38] requires that the objective function takes a certain form involving a strictly convex function and the step size for updating the multipliers is sufficiently small (which is impractical), while no explicit linear rate is given. Its recent update assumes a bounded sequence in addition. On the other hand, it allows more than two blocks of separable variables and it does not require strict convexity; instead, it requires the objective function to include $f(Ex)$, where f is strictly convex and E is a possibly rank-deficient matrix.

The linear convergence of ADMM was also established in the context of Douglas-Rachford Splitting Mehtod (DRSM) [15] and Proximal Point Mehtod (PPA) [50]. It has been shown that ADMM is a special case of DRSM [20], and further DRSM is a special case of PPA [17]. Therefore, the linear convergence of ADMM can be obtained from the existing linear convergence results of DRSM and PPA [40, 50]. However, it is unclear whether those results can apply to the generalizations to ADMM (Algorithm 2). It remains an open question whether these generalizations to ADMM are equivalent to the iterations of some firmly nonexpansive operators, which will be left for future research.

In the next subsection, we introduce a simple technique to slightly improve the result in [35] from $O(1/k)$ to $o(1/k)$. In Chapter 4, we will present detailed analysis to show the linear convergence of the Generalized ADMM under a variety of scenarios in which at least one of the two objective functions is strictly convex and has Lipschitz continuous gradient. This rate is stronger than the sublinear rates such as $O(1/k)$ and $O(1/k^2)$ and is given in terms of the solution error, which is stronger than those given in terms of the objective error in [24, 25, 28], violation of optimality conditions in [36], and the solution relative change in [35]. On the other hand, [24, 25, 35, 36] do not require any strictly convex functions. The fact that a wide range of applications give rise to model (2.6) with at least one strictly convex functions has motivated our work. In Subsection 4.2.3, we will review the linear convergence result of DRSM in [40] and show that our analysis yields a better linear rate, in addition to its wider applicability for generalizations of ADMM.

3.5 On $o(1/k)$ Convergence Rate

In this section, we improve the $O(1/k)$ convergence rate of ADMM established in [35] slightly to $o(1/k)$. The proof technique is based on an elementary lemma (see Lemma 3.2 below) which can be used to improve many other existing $O(1/k)$ convergence rate to $o(1/k)$ as well.

Lemma 3.2. *If a sequence $\{a_k\} \subseteq \mathbb{R}$ obeys: (1) $a_k \geq 0$; (2) $\sum_{k=1}^{\infty} a_k < +\infty$; (3) a_k is monotonically non-increasing, then we have $a_k = o(1/k)$.*

Proof. By the assumptions, we have

$$k \cdot a_{2k} \leq a_{k+1} + a_{k+2} + \cdots + a_{2k} \to 0$$

as $k \to +\infty$. Therefore, $a_k = o(1/k)$. □

Intuitively, the harmonic sequence $\{1/k\}$ is not summable, so a summable, non-negative, monotonic sequence shall converge faster than $\{1/k\}$.

Let us briefly review the analysis in [35]. For simplicity, we consider the classic ADMM (Algorithm 1) but the analysis can be extended to the Generalized ADMM (Algorithm 2) as well. In [35], the quantity $\|u^k - u^{k+1}\|_G^2$ is used to measure the optimality of the iterations, where

$$u := \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}, \quad G := \begin{pmatrix} \beta A^\top A & & \\ & \mathbf{O} & \\ & & \frac{1}{\beta}\mathbf{I} \end{pmatrix}.$$

Note that y is essentially not part of the G-norm. In fact, y can be regarded as an intermediate variable in the iterations of ADMM, whereas x and λ are the essential variables [4]. Based on the optimality conditions (3.20) and (3.21) as well as the relation between $\lambda^k - \lambda^{k+1}$ and $Ax^{k+1} + By^{k+1} - b$, it is easy to see that if $\|u^k - u^{k+1}\|_G^2 =$

0 then u^{k+1} satisfies the KKT conditions and thus is optimal. On the other hand, if $\|u^k - u^{k+1}\|_G^2$ is large, then there is big violation of the KKT conditions so u^{k+1} is very likely far away from being a solution. Therefore, the quantity $\|u^k - u^{k+1}\|_G^2$ can be viewed as a measure of the distance between u^{k+1} and the solution set. Furthermore, it seems reasonable to use the squared term $\|u^k - u^{k+1}\|_G^2$ rather than $\|u^k - u^{k+1}\|_G$ to measure the convergence rate of ADMM. By the variational inequality (3.12) as well as the variational characterization of the ADMM iterations (refer to [35] for more details), we can see that the quadratic term $\|u^k - u^{k+1}\|_G^2$ closely relates to the objective error $f(x^{k+1}) + g(y^{k+1}) - f(x^*) - g(y^*)$. Therefore, in the similar sense that the objective error is commonly used to measure convergence rates, it is reasonable to use the quadratic term $\|u^k - u^{k+1}\|_G^2$ as well.

The work [35] proves that $\|u^k - u^{k+1}\|_G^2$ converges to zero at a rate of $O(1/k)$. The key steps of the proof are to establish the following properties.

Lemma 3.3. *Let $\{u^k\}$ be the sequence generated by Algorithm 1.*

(i) The sequence $\{u^k\}$ is contractive:

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq \|u^k - u^{k+1}\|_G^2. \tag{3.51}$$

(ii) The sequence $\|u^k - u^{k+1}\|_G^2$ is monotonically non-increasing:

$$\|u^k - u^{k+1}\|_G^2 \leq \|u^{k-1} - u^k\|_G^2. \tag{3.52}$$

The contraction property (3.51) follows directly from (3.37). It has been long established in the literature, which dates back to [21,23]. For the sake of completeness, we shall prove the monontonicity property (3.52). Inspired by [35], we provide a much shorter proof than the one in [35].

Proof of (3.52). To simplify the notation, we let

$$\Delta \mathbf{x}^{k+1} = x^k - x^{k+1},\ \Delta \mathbf{y}^{k+1} = y^k - y^{k+1},\ \Delta \lambda^{k+1} = \lambda^k - \lambda^{k+1}.$$

Let us recall the part (i) of Lemma 3.1. We have the following optimality conditions for the subproblems:

$$A^T \lambda^{k+1} \in \partial f(x^{k+1}), \tag{3.53}$$

$$B^T \left(\lambda^{k+1} - \beta A(x^k - x^{k+1})\right) \in \partial g(y^{k+1}). \tag{3.54}$$

By the monotonicity of subdifferential ∂f, combining (3.53) at k-th and $(k+1)$-th iterations yields

$$\langle \Delta \mathbf{x}^{k+1},\ A^\top \Delta \lambda^{k+1} \rangle \geq 0. \tag{3.55}$$

Similarly, using the monotonicity of ∂g for (3.54) at k-th and $(k+1)$-th iterations, we obtain

$$\langle \Delta \mathbf{y}^{k+1},\ B^\top \Delta \lambda^{k+1} - \beta B^\top A(\Delta \mathbf{x}^k - \Delta \mathbf{x}^{k+1}) \rangle \geq 0. \tag{3.56}$$

Adding the above two inequalities together, we have

$$(A\Delta \mathbf{x}^{k+1} + B\Delta \mathbf{y}^{k+1})^\top \Delta \lambda^{k+1} - \beta (B\Delta \mathbf{y}^{k+1})^\top A(\Delta \mathbf{x}^k - \Delta \mathbf{x}^{k+1}) \geq 0. \tag{3.57}$$

Note that we have

$$A\Delta \mathbf{x}^{k+1} + B\Delta \mathbf{y}^{k+1} = \frac{1}{\beta}(\Delta \lambda^k - \Delta \lambda^{k+1}). \tag{3.58}$$

Then we substitute $\Delta \mathbf{y}^{k+1}$ using (3.58), and thereby (3.57) becomes

$$\frac{1}{\beta}(\Delta \lambda^k - \Delta \lambda^{k+1})^\top \Delta \lambda^{k+1} - (\Delta \lambda^k - \Delta \lambda^{k+1} - \beta A \Delta \mathbf{x}^{k+1})^\top A(\Delta \mathbf{x}^k - \Delta \mathbf{x}^{k+1}) \geq 0. \tag{3.59}$$

To further simplify the notation, we let

$$\mathbf{a}^k = \sqrt{\beta} A \Delta \mathbf{x}^k, \ \mathbf{b}^k = \frac{1}{\sqrt{\beta}} \Delta \lambda^k.$$

After rearranging the terms, (3.59) can be rewritten as

$$\begin{aligned} &(\mathbf{a}^k + \mathbf{b}^k)^\top (\mathbf{a}^{k+1} + \mathbf{b}^{k+1}) - (\mathbf{a}^k)^\top \mathbf{b}^k - (\mathbf{a}^{k+1})^\top \mathbf{b}^{k+1} \\ &\geq \|\mathbf{a}^{k+1}\|^2 + \|\mathbf{b}^{k+1}\|^2 = \|u^k - u^{k+1}\|_G^2. \end{aligned} \tag{3.60}$$

By the Cauchy-Schwarz inequality, we have

$$(\mathbf{a}^k + \mathbf{b}^k)^\top (\mathbf{a}^{k+1} + \mathbf{b}^{k+1}) \leq (\|\mathbf{a}^k + \mathbf{b}^k\|^2 + \|\mathbf{a}^{k+1} + \mathbf{b}^{k+1}\|^2)/2, \tag{3.61}$$

or equivalently,

$$\begin{aligned} &(\mathbf{a}^k + \mathbf{b}^k)^\top (\mathbf{a}^{k+1} + \mathbf{b}^{k+1}) - (\mathbf{a}^k)^\top \mathbf{b}^k - (\mathbf{a}^{k+1})^\top \mathbf{b}^{k+1} \\ &\leq (\|\mathbf{a}^k\|^2 + \|\mathbf{b}^k\|^2 + \|\mathbf{a}^{k+1}\|^2 + \|\mathbf{b}^{k+1}\|^2)/2. \end{aligned} \tag{3.62}$$

Applying the above inequality to the left-hand side of (3.60), we have

$$\begin{aligned} \|u^k - u^{k+1}\|_G^2 &\leq (\|\mathbf{a}^k\|^2 + \|\mathbf{b}^k\|^2 + \|\mathbf{a}^{k+1}\|^2 + \|\mathbf{b}^{k+1}\|^2)/2 \\ &= \left(\|u^{k-1} - u^k\|_G^2 + \|u^k - u^{k+1}\|_G^2\right)/2, \end{aligned} \tag{3.63}$$

and thus (3.52) follows immediately. □

We are now ready to improve the convergence rate from $O(1/k)$ to $o(1/k)$.

Theorem 3.3 (Convergence Rate of $o(1/k)$)**.** *Let $\{u^k\}$ be the sequence generated by Algorithm 1. Then $\|u^k - u^{k+1}\|_G^2 = o(1/k)$. Therefore,*

- $\|Ax^k - Ax^{k+1}\|^2 + \|By^k - By^{k+1}\|^2 + \|\lambda^k - \lambda^{k+1}\|^2 = o(1/k)$;
- $\|Ax^{k+1} + By^{k+1} - b\|^2 = o(1/k)$.

Proof. By (3.51), we have

$$\sum_{k=1}^{n} \|u^k - u^{k+1}\|_G^2 \leq \|u^1 - u^*\|_G^2 - \|u^{n+1} - u^*\|_G^2, \ \forall n. \tag{3.64}$$

As we have shown, $\|u^{n+1} - u^*\|_G^2 \to 0$ as $n \to \infty$. Therefore, $\sum_{k=1}^{\infty} \|u^k - u^{k+1}\|_G^2 < \infty$.

By (3.52), $\|u^k - u^{k+1}\|_G^2$ is monotonically non-increasing. Obviously, $\|u^k - u^{k+1}\|_G^2$ is also nonnegative. Therefore, $\|u^k - u^{k+1}\|_G^2 = o(1/k)$ follows from Lemma 3.2 immediately. By the definition of G, we have $\|Ax^k - Ax^{k+1}\|^2 = o(1/k)$ and $\|\lambda^k - \lambda^{k+1}\|^2 = o(1/k)$. By (3.58), we have $\|By^k - By^{k+1}\|^2 = o(1/k)$ as well. Finally, $\|Ax^{k+1} + By^{k+1} - b\|^2 = \|\lambda^k - \lambda^{k+1}\|^2/\beta^2 = o(1/k)$. □

Remark 3.3. *The proof technique based on Lemma 3.2 can be applied to improve some other existing convergence rates of $O(1/k)$ (e.g., [12, 33]) to $o(1/k)$ as well.*

Chapter 4

Linear Convergence of Generalized ADMM

The main goal of this chapter is to show that the Generalized ADMM (Algorithm 2) applied to the problem (2.6) has global linear convergence $O(1/c^k)$ for some $c > 1$, provided that one of the two objective functions, say, f is *strictly* convex, ∇f is Lipschitz continuous, and matrices A and B have certain rank conditions. Note that when restricted to a compact set, a strictly convex function is strongly convex. So, as long as an algorithm generates a bounded sequence, strict convexity is effectively strong convexity. For simplicity, we use "strong convexity" or "strongly convex" in the remaining of the chapter.

The chapter is organized as follows. Section 4.1 summarizes the linear convergence results that we shall establish under various scenarios. The detailed analysis is then given in Section 4.2. Section 4.3 discusses several interesting applications that are covered by our linear convergence results. Section 4.4 presents numerical results to demonstrate the linear convergence behavior of ADMM in practice.

4.1 Summary of Results

We shall establish the global linear convergence of Algorithm 2 that are described in Tables 4.1 and 4.2. Table 4.1 summarizes the four scenarios under which we study the linear convergence of Algorithm 2, and Table 4.2 specifies the linear convergent

Table 4.1 : Four scenarios leading to linear convergence

scenario	strongly convex	Lipschitz continuous	full row rank	additional assumptions
1	f	∇f	A	*if* $Q \succ 0$, B has full column rank
2	f, g	∇f	A	
3	f	$\nabla f, \nabla g$	-	B has full column rank
4	f, g	$\nabla f, \nabla g$	-	

quantities for different types of matrices $\hat{P}$, P, and Q, where

$$\hat{P} := P + \beta A^T A$$

is defined for the convenience of convergence analysis. $P = 0$ and $Q = 0$ correspond to exactly solving the x- and y-subproblems, respectively. Although $P = 0$ and $\hat{P} \succ 0$ are different cases in Table 4.2, they may happen at the same time if A has full column rank; if so, apply the result under $\hat{P} \succ 0$, which is stronger.

The conclusions in Table 4.2 are the quantities that converge either Q-linearly or R-linearly*. Q-linear convergent quantities are the *entireties* of multiple variables whereas R-linear convergent quantities are the individual variables x^k, y^k, and λ^k.

Four scenarios of global linear convergence. In scenario 1, only function f needs to be strongly convex and having Lipschitz continuous gradient; there is no

*Suppose a sequence $\{u^k\}$ converges to u^*. We say the convergence is (in some norm $\|\cdot\|$)

- *Q-linear*, if there exists $\mu \in (0, 1)$ such that $\frac{\|u^{k+1}-u^*\|}{\|u^k-u^*\|} \leq \mu$;
- *R-linear*, if there exists a sequence $\{\sigma^k\}$ such that $\|u^k - u^*\| \leq \sigma^k$ and $\sigma^k \to 0$ Q-linearly.

Table 4.2 : Summary of linear convergence results

case	$P, \hat{P}$	Q	any scenario 1 – 4	
			Q-linear convergence	R-linear convergence
1	$P = 0$	$= 0$	(Ax^k, λ^k)	x^k, $(y^k$ or $By^k)^*$, λ^k
2	$\hat{P} \succ 0$	$= 0$	(x^k, λ^k)	
3	$P = 0$	$\succ 0$	(Ax^k, y^k, λ^k)	
4	$\hat{P} \succ 0$	$\succ 0$	(x^k, y^k, λ^k)	

* In cases 1 and 2, scenario 1, R-linear convergence of y^k requires full column rank of B; otherwise, only By^k has R-linear convergence.

assumption on g besides convexity. On the other hand, matrix A must have full row rank. Roughly speaking, the full row rank of A makes sure that the error of λ^k can be bounded just from the x-side by applying the Lipschitz continuity of ∇f. One cannot remove this condition or relax it to the full row rank of $[A, B]$ without additional assumptions. Consider the example of $A = [1; 0]$ and $B = [0; 1]$, where $[A, B] = I$ has full rank. Since λ_2^k, which is the 2nd entry of λ^k, is not affected by f or $\{x^k\}$ at all, there is no way to take advantages of the Lipschitz continuity of ∇f to bound the error of λ_2^k. In general, without the full row rank of A, a part of λ^k needs to be controlled from the y-side using properties of g.

Scenario 2 adds the strong convexity assumption on g. As a result, the remark in case 1 regarding the full column rank of B is no longer needed.

Both scenarios 3 and 4 assume that g is differentiable and ∇g is Lipschitz continuous. As a result, the error of λ^k can be controlled by taking advantages of the Lipschitz continuity of both ∇f and ∇g, and the full row rank assumption on A is

no longer needed. On the other hand, scenarios 3 and 4 exclude the problems with non-differentiable g. Compared to scenario 3, scenario 4 adds the strong convexity assumption on g and drops the remark on the full column rank of B.

Under scenario 1 with $Q \succ 0$ and scenario 3, the remarks in Table 4.1 are needed essentially because y^k gets coupled with x^k and λ^k in certain inequalities in our convergence analysis. The full column rank of B helps bound the error of y^k by those of x^k and λ^k.

Four cases. When $P = 0$ (corresponding to exactly solving the x-subproblem), we have $\hat{P} \succeq 0$ and only obtain linear convergence in Ax. However, when $\hat{P} \succ 0$, linear convergence in x is obtained.

When $Q = 0$ (corresponding to exactly solving the y-subproblem), y is not part of the Q-linear convergent joint variable. But, when $Q \succ 0$, y becomes part of it.

Remark 4.1. *Indeed, we only use f and g's properties over the compact sets including $\{x^k\}$ and $\{y^k\}$, not globally; thus, strict convexity can replace strong convexity.*

4.2 Linear Convergence

To establish the linear convergence, we take three steps. First, using (3.34) for $P \neq 0$ and (3.36) for $P = 0$, as well as the assumptions in Table 4.1, we show that there exists $\delta > 0$ such that

$$\|u^k - u^*\|_G^2 \geq (1 + \delta)\|u^{k+1} - u^*\|_G^2, \tag{4.1}$$

where $u^* = \lim_{k\to\infty} u^k$ is given by Theorem 3.2. We call (4.1) the Q-linear convergence of $\{u^k\}$ in G-(semi)norm. Next, using (4.1) and the definition of G, we obtain the Q-linear convergent quantities in Table 4.2. Finally, the R-linear convergence in Table 2 is established.

4.2.1 Analysis

We first assume $\gamma = 1$, which allows us to simplify the proof presentation. At the end of this subsection, we explain why the results for $\gamma = 1$ can be extended to $\gamma \neq 1$ that satisfies the conditions of Theorem 3.1. Note that for $\gamma = 1$, we have (3.37) instead of (3.36). Hence, no matter $P = 0$ or $P \neq 0$, both inequalities (3.34) and (3.37) have the form

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq C,$$

where C stands for their right-hand sides. To show (4.1), it is sufficient to establish

$$C \geq \delta \|u^{k+1} - u^*\|_G^2. \tag{4.2}$$

The challenge is that $\|u^{k+1} - u^*\|_G^2$ is the sum of $\|x^{k+1} - x^*\|_{\hat{P}}^2$, $\|y^{k+1} - y^*\|_Q^2$, and $\frac{1}{\beta\gamma}\|\lambda^{k+1} - \lambda^*\|^2$, but C does not contain terms like $\|y^{k+1} - y^*\|^2$ and $\|\lambda^{k+1} - \lambda^*\|^2$. Therefore, we shall bound $\|\lambda^{k+1} - \lambda^*\|^2$ and $\|y^{k+1} - y^*\|_Q^2$ from the existing terms in C or using the strong convexity assumptions. This is done in a series of lemmas below.

Lemma 4.1 (For scenario 1, cases 3 and 4, and scenario 3)**.** *Suppose that B has full column rank. For any $\mu_1 > 0$, we have*

$$\|y^{k+1} - y^*\|^2 \leq c_1 \|x^{k+1} - x^*\|^2 + c_2 \|\lambda^k - \lambda^{k+1}\|^2, \tag{4.3}$$

where $c_1 := (1 + \frac{1}{\mu_1})\|A\|^2 \cdot \lambda_{\min}^{-1}(B^T B) > 0$ and $c_2 := (1 + \mu_1)(\beta\gamma)^{-2} \cdot \lambda_{\min}^{-1}(B^T B) > 0$.

Proof. By (3.24), we have $\|B(y^{k+1} - y^*)\|^2 = \|A(x^{k+1} - x^*) - \frac{1}{\beta\gamma}(\lambda^k - \lambda^{k+1})\|^2$. Then apply the following inequality (or the Cauchy-Schwarz inequality):

$$\|u + v\|^2 \leq \left(1 + \frac{1}{\mu_1}\right)\|u\|^2 + (1 + \mu_1)\|v\|^2, \ \forall \mu_1 > 0, \tag{4.4}$$

to its right-hand side. □

Lemma 4.2 (For scenarios 1 and 2)**.** *Suppose that ∇f is Lipschitz continuous with constant L_f and A has full row rank. For any $\mu_2 > 1$, we have*

$$\|\hat{\lambda} - \lambda^*\|^2 \leq c_3\|x^{k+1} - x^*\|^2 + c_4\|x^k - x^{k+1}\|^2, \tag{4.5}$$

where $c_3 := L_f^2(1 - \frac{1}{\mu_2})^{-1}\lambda_{\min}^{-1}(AA^T) > 0$ and $c_4 := \mu_2\|P\|^2\lambda_{\min}^{-1}(AA^T) > 0$.

Proof. By the optimality conditions (3.9) and (3.20) together with the Lipschitz continuity of ∇f, we have

$$\|A^T(\hat{\lambda} - \lambda^*) + P(x^k - x^{k+1})\|^2 = \|\nabla f(x^{k+1}) - \nabla f(x^*)\|^2 \leq L_f^2\|x^{k+1} - x^*\|^2. \tag{4.6}$$

Then apply the following basic inequality:

$$\|u + v\|^2 \geq \left(1 - \frac{1}{\mu_2}\right)\|u\|^2 + (1 - \mu_2)\|v\|^2, \ \forall \mu_2 > 0, \tag{4.7}$$

to the left hand side of (4.6). We require $\mu_2 > 1$ so that $(1 - \frac{1}{\mu_2}) > 0$. □

Lemma 4.3 (For scenarios 3 and 4)**.** *Suppose ∇f and ∇g are Lipschitz continuous, and the initial multiplier λ^0 is in the range space of $[A, B]$ (letting $\lambda^0 = 0$ suffices). For any $\mu_3 > 1$ and $\mu_4 > 0$, we have*

$$\|\hat{\lambda} - \lambda^*\|^2 \leq c_5\|x^k - x^{k+1}\|^2 + c_6\|y^k - y^{k+1}\|_Q^2 + c_7\|x^{k+1} - x^*\|^2 + c_8\|y^{k+1} - y^*\|^2, \tag{4.8}$$

where $c_5 = \mu_3(1 + \frac{1}{\mu_4})\|[P^T, -\beta A^T B]\|^2\bar{c} > 0$, $c_6 = \mu_3(1 + \mu_4)\|Q\|^2\bar{c} \geq 0$, $c_7 = (1 - \frac{1}{\mu_3})^{-1}L_f^2\bar{c} > 0$, $c_8 = (1 - \frac{1}{\mu_3})^{-1}L_g^2\bar{c} > 0$, and $\bar{c} > 0$ is as follows:

- *If the matrix $[A, B]$ has full row rank, $\bar{c} := \lambda_{\min}^{-1}([A, B][A, B]^T) > 0$.*
- *Otherwise, rank$([A, B]) = r < p$. Without loss of generality, assuming the first r rows of $[A, B]$ (denoted by $[A_r, B_r]$) are linearly independent, we have*

$$[A, B] = \begin{bmatrix} I \\ L \end{bmatrix} [A_r, B_r],$$

where $I \in \mathbb{R}^{r\times r}$ *is the identity matrix and* $L \in \mathbb{R}^{(p-r)\times r}$. *Let* $E := (I + L^T L)[A_r, B_r]$, *and* $\bar{c} := \lambda_{\min}^{-1}(EE^T)\|I + L^T L\| > 0$.

Proof. We first show that

$$\|\hat{\lambda} - \lambda^*\|^2 \le \bar{c} \cdot \left\| \begin{bmatrix} A^T \\ B^T \end{bmatrix} (\hat{\lambda} - \lambda^*) \right\|^2, \tag{4.9}$$

where $\bar{c} > 0$ is defined above. If $[A, B]$ has full row rank, it is trivial. Now, suppose $[A, B]$ is rank deficient, i.e., $\text{rank}([A, B]) = r < p$. Without loss of generality, we assume the first r rows of $[A, B]$ (denoted by $[A_r, B_r]$) are linearly independent, and thus

$$[A, B] = \begin{bmatrix} I \\ L \end{bmatrix} [A_r, B_r].$$

By the update formula, if the initial multiplier λ^0 is in the range space of $[A, B]$, then λ^k, $k = 1, 2, \ldots$, always stay in the range space of $[A, B]$, so do $\hat{\lambda}$ and λ^*. It follows that

$$\lambda^k = \begin{bmatrix} I \\ L \end{bmatrix} \lambda_r^k, \ \hat{\lambda} = \begin{bmatrix} I \\ L \end{bmatrix} \hat{\lambda}_r, \ \lambda^* = \begin{bmatrix} I \\ L \end{bmatrix} \lambda_r^*.$$

and thus

$$\begin{bmatrix} A^T \\ B^T \end{bmatrix} (\hat{\lambda} - \lambda^*) = \begin{bmatrix} A_r^T \\ B_r^T \end{bmatrix} (I + L^T L)(\hat{\lambda}_r - \lambda_r^*).$$

Since $E := (I + L^T L)[A_r, B_r]$ has full row rank, we have $\bar{c} := \lambda_{\min}^{-1}(EE^T)\|I + L^T L\| > 0$ and (4.9) follows immediately.

Combining the optimality conditions (3.10), (3.9), (3.20), and (3.21) together with

the Lipschitz continuity of ∇f and ∇g, we have

$$\begin{aligned}
&\left\| \begin{bmatrix} A^T \\ B^T \end{bmatrix} (\hat{\lambda} - \lambda^*) + \begin{bmatrix} P \\ -\beta B^T A \end{bmatrix} (x^k - x^{k+1}) + \begin{bmatrix} \mathbf{0} \\ Q \end{bmatrix} (y^k - y^{k+1}) \right\|^2 \\
=&\|\nabla f(x^{k+1}) - \nabla f(x^*)\|^2 + \|\nabla g(y^{k+1}) - \nabla g(y^*)\|^2 \\
\leq& L_f^2 \|x^{k+1} - x^*\|^2 + L_g^2 \|y^{k+1} - y^*\|^2.
\end{aligned} \tag{4.10}$$

Similarly, we apply the basic inequalities (4.4) and (4.7) to its left hand side and use (4.9). □

With the above lemmas, we now prove the following main theorem of this subsection.

Theorem 4.1 (Q-linear convergence of $\|u^k - u^*\|_G^2$)**.** *Under the same assumptions of Theorem 3.2 and $\gamma = 1$, for all scenarios in Table 4.1, there exists $\delta > 0$ such that (4.1) holds.*

Proof. Consider the case of $P = 0$ and the corresponding inequality (3.37). In this case $\hat{P} = \beta A^T A \succeq 0$. Let C denote the right-hand side of (3.37).

Scenarios 1 and 2 (recall in both scenarios, f is strongly convex, ∇f is Lipschitz continuous, and A has full row rank). Note that C contains the terms on the right side of (4.5) with strictly positive coefficients. Hence, applying Lemma 4.2 to C, we can obtain

$$\begin{aligned}
C \geq& (c_9 \|x^{k+1} - x^*\|^2 + c_{10} \|y^{k+1} - y^*\|^2 + c_{11} \|\lambda^{k+1} - \lambda^*\|^2) \\
&+ (c_{12} \|y^k - y^{k+1}\|_Q^2 + c_{13} \|\lambda^k - \lambda^{k+1}\|^2)
\end{aligned} \tag{4.11}$$

with $c_9, c_{11} > 0$, $c_{10} = 2\nu_g \geq 0$, $c_{12} = \eta > 0$, and $c_{13} = \eta/(\beta\gamma) > 0$. We have $c_9 > 0$ because only a fraction of $2\nu_f \|x^{k+1} - x^*\|^2$ is used with Lemma 4.2; $c_9 \|x^{k+1} - x^*\|^2$ is unused so it stays. The same principle is applied below to get strictly positive

coefficients, and we do not re-state it. For proof brevity, we do not necessarily specify the values of c_i.

For scenario 1 with $Q = 0$, $\|u^{k+1} - u^*\|_G^2 = \|x^{k+1} - x^*\|_{\hat{P}}^2 + \frac{1}{\beta\gamma}\|\lambda^{k+1} - \lambda^*\|^2$. Since $\|x^{k+1} - x^*\|^2 \geq \lambda_{\max}(\hat{P})^{-1}\|x^{k+1} - x^*\|_{\hat{P}}^2$, (4.2) follows from (4.11) with $\delta = \min\{c_9\lambda_{\max}^{-1}(\hat{P}), c_{11}\beta\gamma\} > 0$.

For scenario 1 with $Q \succ 0$, $\|u^{k+1}-u^*\|_G^2 = \|x^{k+1}-x^*\|_{\hat{P}}^2+\|y^{k+1}-x^*\|_Q^2+\frac{1}{\beta\gamma}\|\lambda^{k+1}-\lambda^*\|^2$. Since c_{10} is not necessarily strictly positive, we shall apply Lemma 4.1 to (4.11) and obtain

$$C \geq (c_{14}\|x^{k+1} - x^*\|^2 + c_{15}\|y^{k+1} - y^*\|^2 + c_{11}\|\lambda^{k+1} - \lambda^*\|^2) + c_{12}\|y^k - y^{k+1}\|_Q^2 \quad (4.12)$$

where $c_{14}, c_{15}, c_{11}, c_{12} > 0$. So, it leads to (4.1) with

$$\delta = \min\{c_{14}\lambda_{\max}^{-1}(\hat{P}), c_{15}\lambda_{\max}^{-1}(Q), c_{11}\beta\gamma\} > 0.$$

Scenario 2 (recall it is scenario 1 plus that g is strongly convex). We have $c_{10} = 2\nu_g > 0$ in (4.11), which gives (4.1) with $\delta = \min\{c_9\lambda_{\max}^{-1}(\hat{P}), c_{10}\lambda_{\max}^{-1}(Q), c_{11}\beta\gamma\} > 0$. Note that we have used the convention that if $Q = 0$, then $\lambda_{\max}^{-1}(Q) = \infty$.

Scenario 3 (recall f is strongly convex, both ∇f and ∇g are Lipschitz continuous). We apply Lemma 4.1 to get $\|y^{k+1} - y^*\|^2$ with which we then apply Lemma 4.3 to obtain

$$C \geq c_{16}\|x^{k+1} - x^*\|^2 + c_{17}\|y^{k+1} - y^*\|^2 + c_{18}\|\lambda^{k+1} - \lambda^*\|^2, \quad (4.13)$$

where $c_{16}, c_{17}, c_{18} > 0$ and the terms $\|x^k - x^{k+1}\|^2$, $\|y^k - y^{k+1}\|^2$, and $\|\lambda^k - \lambda^{k+1}\|^2$ with nonnegative coefficients have been dropped from the right-hand side of (4.13). From (4.13), we obtain (4.1) with $\delta = \min\{c_{16}\lambda_{\max}^{-1}(\hat{P}), c_{17}\lambda_{\max}^{-1}(Q), c_{18}\beta\gamma\} > 0$.

Scenario 4 (recall it is scenario 3 plus that g is strongly convex). Since $c_{11} = 2\nu_g > 0$ in (4.11), we can directly apply Lemma 4.3 to get (4.1) with $\delta > 0$ in a way similar to scenario 3.

Now consider the case of $P \neq 0$ and the corresponding inequality (3.32). Inequalities (3.32) and (3.37) are similar except (3.37) has the extra term $\|x^k - x^{k+1}\|^2$ with a strictly positive coefficient in its right-hand side. This term is needed when Lemma 4.2 is applied. However, the assumptions of the theorem ensure $\hat{P} \succ 0$ whenever $P \neq 0$. Therefore, in (3.32), the term $\|u^k - u^{k+1}\|_G^2$, which contains $\|x^k - x^{k+1}\|_{\hat{P}}^2$, can spare out a term $c_{19}\|x^k - x^{k+1}\|^2$ with $c_{19} > 0$. Therefore, following the same arguments for the case of $P = 0$, we get (4.1) with certain $\delta > 0$. □

Now we extend the result in Theorem 4.1 (which is under $\gamma = 1$) to $\gamma \neq 1$ in the following theorem.

Theorem 4.2. *Under the same assumptions of Theorem 3.2 and $\gamma \neq 1$, for all scenarios in Table 4.1,*

i) if $P \neq 0$, there exists $\delta > 0$ such that (4.1) holds;

ii) if $P = 0$, there exists $\delta > 0$ such that

$$\|u^k - u^*\|_G^2 + \frac{\beta}{\rho}\|r^k\|^2 \geq (1+\delta)\left(\|u^{k+1} - u^*\|_G^2 + \frac{\beta}{\rho}\|r^{k+1}\|^2\right). \tag{4.14}$$

Proof. When $\gamma \neq 1$, which causes $\lambda^{k+1} \neq \hat{\lambda}$. We shall bound $\|\lambda^{k+1} - \lambda^*\|^2$ but Lemmas 4.2 and 4.3 only give bounds on $\|\hat{\lambda} - \lambda^*\|^2$. Noticing that $(\hat{\lambda} - \lambda^*) - (\lambda^{k+1} - \lambda^*) = \hat{\lambda} - \lambda^{k+1} = (\gamma - 1)r^{k+1}$ and C contains a strictly positive term in $\|\lambda^k - \lambda^{k+1}\|^2 = \gamma^2\|r^{k+1}\|^2$, we can bound $\|\lambda^{k+1} - \lambda^*\|^2$ by a positively weighted sum of $\|\hat{\lambda} - \lambda^*\|^2$ and $\|\lambda^k - \lambda^{k+1}\|^2$.

If $P \neq 0$, the rest of the proof follows from that of Theorem 4.1.

If $P = 0$, $\gamma \neq 1$ leads to (3.36), which extends $\|u^i - u^*\|_G^2$ in (3.37) to $\|u^i - u^*\|_G^2 + \frac{\beta}{\rho}\|r^i\|^2$, for $i = k, k+1$. Since C contains $\|\lambda^k - \lambda^{k+1}\|^2 = \gamma^2\|r^{k+1}\|^2$ with a strictly positive coefficient, one obtains (4.14) by using this term and following the proof of Theorem 4.1. □

Q-linear convergent quantities

From the definition of G, which depends on P and Q, it is easy to see that the Q-linear convergence of $u^k = (x^k; y^k; \lambda^k)$ translates to the Q-linear convergence results in Table 4.2. For example, in case 1 ($P = 0$ and $Q = 0$), $\|u^{k+1} - u^*\|_G^2 = \|x^{k+1} - x^*\|_{\hat{P}}^2 + \frac{1}{\beta\gamma}\|\lambda^{k+1} - \lambda^*\|^2$, where $\hat{P} = P + \beta AA^T = \beta A^T A$. Hence, (Ax^k, λ^k) converges Q-linearly. Examining $\|u^{k+1} - u^*\|_G^2$ gives the results for cases 2, 3, 4.

R-linear convergent quantities

By the definition of R-linear convergence, any part of a Q-linear convergent quantity converges R-linearly. For example, in case 1 ($P = 0$ and $Q = 0$), the Q-linear convergence of (Ax^k, λ^k) in Table 4.2 gives the R-linear convergence of Ax^k and λ^k. Therefore, to establish Table 4.2, it remains to show the R-linear convergence of x^k in cases 1 and 3 and that of y^k in cases 1 and 2. Our approach is to bound their errors by existing R-linear convergent quantities.

Theorem 4.3 (R-linear convergence)**.** *The following statements hold.*

i) In cases 1 and 3, if λ^k converges R-linearly, then x^k converges R-linearly.

ii) In cases 1 and 2, scenario 1, if λ^k and x^k both converge R-linearly, then By^k converges R-linearly. In addition, if B has full column rank, then y^k converges R-linearly.

iii) In cases 1 and 2, scenarios 2–4, if λ^k and x^k both converge R-linearly, then y^k converges R-linearly.

Proof. We only show the result for $\gamma = 1$ (thus $\hat{\lambda} = \lambda^{k+1}$); for $\gamma \neq 1$ (thus $\hat{\lambda} \neq \lambda^{k+1}$), the results follow from those for $\gamma = 1$ and the R-linear convergence of $\|\hat{\lambda} - \lambda^{k+1}\|^2$,

which itself follows from (3.16) and the R-linear convergence of λ^k (thus that of $\lambda^k - \lambda^{k+1}$).

i) By (3.22) and $\hat{P} = \beta A^T A$, we have $\nu_f \|x^{k+1} - x^*\|^2 \leq \|A\| \|x^{k+1} - x^*\| \|\lambda^{k+1} - \lambda^*\|$, which implies

$$\|x^{k+1} - x^*\|^2 \leq \frac{\|A\|^2}{\nu_f^2} \|\lambda^{k+1} - \lambda^*\|^2. \tag{4.15}$$

ii) The result follows from (3.24).

iii) Scenario 3 assumes the full column rank of B, so the result follows from (3.24). In scenarios 2 and 4, g is strongly convex. Recall (3.23) with $\hat{\lambda} = \lambda^{k+1}$:

$$\langle y^{k+1} - y^*,\ B^T \left(\lambda^{k+1} - \lambda^* - \beta A(x^k - x^{k+1})\right) + Q(y^k - y^{k+1}) \rangle \geq \nu_g \|y^{k+1} - y^*\|^2. \tag{4.16}$$

By the Cauchy-Schwarz inequality and $Q = \mathbf{0}$, we have

$$\nu_g \|y^{k+1} - y^*\| \leq \|B\| \|\lambda^{k+1} - \lambda^* - \beta A(x^k - x^{k+1})\|. \tag{4.17}$$

Therefore, the result follows from the R-linear convergence of x^k and λ^k.

□

4.2.2 Explicit Formula of Linear Rate

To keep the proof of Theorem 4.1 easy to follow, we have avoided giving the explicit formulas of c_i's and thus also those of δ. To give the reader an idea what quantities affect δ, we now provide an explicit formula of δ for the classic ADMM (i.e., case 1 with $\gamma = 1$) under scenario 1.

Corollary 4.1 (Convergence rate of classic ADMM under scenario 1)**.** *Under Assumptions 3.1 and 3.2, for scenario 1 in Table 4.1, the sequence* $\{u^k\}$ *of Algorithm 1 satisfies* (4.1) *with*

$$\delta = 2\left(\frac{\beta\|A\|^2}{\nu_f} + \frac{L_f}{\beta\lambda_{\min}(AA^T)}\right)^{-1}. \tag{4.18}$$

In particular, choosing $\beta = \sqrt{\frac{L_f\nu_f}{\|A\|^2\lambda_{\min}(AA^T)}}$ *yields the largest* δ*:*

$$\delta_{\max} = \frac{1}{\kappa_A\sqrt{\kappa_f}}, \tag{4.19}$$

where $\kappa_A := \sqrt{\lambda_{\max}(AA^T)/\lambda_{\min}(AA^T)}$ *is the condition number of matrix* A*, and* $\kappa_f = L_f/\nu_f$ *is the condition number of function* f*.*

Proof. Recall the important inequality (3.37) in Theorem 3.1:

$$\begin{aligned}&\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2\\ \geq& 2\nu_f\|x^{k+1} - x^*\|^2 + 2\nu_g\|y^{k+1} - y^*\|^2 + \|u^k - u^{k+1}\|_G^2 + 2\nu_f\|x^k - x^{k+1}\|^2.\end{aligned} \tag{4.20}$$

Note that the term $\nu_f\|x^{k+1} - x^*\|^2$ on the right-hand side comes from (3.22):

$$\langle x^{k+1} - x^*,\ A^T(\lambda^{k+1} - \lambda^*)\rangle \geq \nu_f\|x^{k+1} - x^*\|^2, \tag{4.21}$$

due to the strong convexity of f and the optimality conditions:

$$A^T\lambda^{k+1} = \nabla f(x^{k+1}),\ A^T\lambda^* = \nabla f(x^*).$$

On the other hand, since ∇f is Lipschitz continuous and A has full row rank, using (2.34) yields

$$\langle x^{k+1} - x^*,\ A^T(\lambda^{k+1} - \lambda^*)\rangle \geq \frac{1}{L_f}\|A^T(\lambda^{k+1} - \lambda^*)\|^2 \geq \frac{\lambda_{\min}(AA^T)}{L_f}\|\lambda^{k+1} - \lambda^*\|^2. \tag{4.22}$$

By combining (4.21) and (4.22), it follows that for any $t \in [0, 1]$,

$$\langle x^{k+1} - x^*,\ A^T(\lambda^{k+1} - \lambda^*)\rangle \geq t\cdot\nu_f\|x^{k+1} - x^*\|^2 + (1-t)\frac{\lambda_{\min}(AA^T)}{L_f}\|\lambda^{k+1} - \lambda^*\|^2. \tag{4.23}$$

Now, using (4.23) to replace (4.21) in our analysis in Section 3.3, the inequality (4.20) can be further refined as

$$\begin{aligned}
&\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \\
\geq\ & 2t \cdot \nu_f \|x^{k+1} - x^*\|^2 + 2(1-t)\frac{\lambda_{\min}(AA^T)}{L_f}\|\lambda^{k+1} - \lambda^*\|^2 \\
&+ 2\nu_g \|y^{k+1} - y^*\|^2 + \|u^k - u^{k+1}\|_G^2 + 2\nu_f \|x^k - x^{k+1}\|^2,\ \forall t \in [0,1].
\end{aligned} \tag{4.24}$$

In particular, letting

$$t = \left(1 + \frac{L_f \nu_f}{\beta^2 \|A\|^2 \lambda_{\min}(AA^T)}\right)^{-1}, \tag{4.25}$$

we have

$$\begin{aligned}
&2t \cdot \nu_f \|x^{k+1} - x^*\|^2 + 2(1-t)\frac{\lambda_{\min}(AA^T)}{L_f}\|\lambda^{k+1} - \lambda^*\|^2 \\
\geq\ & \delta \left(\beta \|A\|^2 \|x^{k+1} - x^*\|^2 + \frac{1}{\beta}\|\lambda^{k+1} - \lambda^*\|^2\right) \geq \delta \|u^{k+1} - u^*\|_G^2,
\end{aligned} \tag{4.26}$$

where $\delta > 0$ is given by (4.18). Then (4.1) follows from (4.24) and (4.26) immediately.

□

Not surprisingly, the convergence rate under scenario 1 is negatively affected by the condition numbers of A and f. For other scenarios, the formulas of δ can also be similarly obtained by deriving the specific values of c_i's in our analysis. However, they appear to be more complicated than the nice formula (4.19) for scenario 1. A close look at these formulas of c_i's reveals that the convergence rate is negatively affected by the condition numbers of the constraint matrices A, B and $[A, B]$, as well as the condition numbers of the objective functions f and g. Due to page limit, we leave other scenarios/cases and further analysis to future research.

Remark 4.2. *It is well known that the penalty parameter β can significantly affect the speed of ADMM. Since the rate of convergence developed in this section is a function*

of β, the rate can be optimized over β which sheds some lights on how to choose a "good" β. More analysis and numerical simulations are left as future research.

4.2.3 Comparison with Linear Rate of DRSM

It is known that applying the classic ADMM to problem (2.6) is equivalent to applying the Douglas-Rachford splitting method (DRSM) to the dual of (2.6). (However, it is unclear to which splitting methods the various ADMM generalizations correspond to.) In this subsection, we review the classic linear convergence result [40] of DRSM. In comparison, we show that our linear rate for the classic ADMM is considerably better than the one in [40].

The dual of (2.6) is given by

$$\min_{\lambda} \ \left\{ -\min_{x,y} \ f(x) + g(y) - \lambda^\top (Ax + By - b) \right\} = \min_{\lambda} \ f^*(A^\top \lambda) + g^*(B^\top \lambda) - b^\top \lambda, \tag{4.27}$$

where f^* and g^* are the *convex conjugate* functions of f and g, respectively. Define the *maximal monotone operators* $\mathcal{A}$ and $\mathcal{B}$ as follows:

$$\mathcal{A}(\lambda) := \partial[g^*(B^\top \lambda)] - b, \quad \mathcal{B}(\lambda) := \partial[f^*(A^\top \lambda)]. \tag{4.28}$$

Then (4.27) is equivalent to finding a zero of the sum of two maximal monotone operators:

$$0 \in \mathcal{A}(\lambda) + \mathcal{B}(\lambda). \tag{4.29}$$

Applying DRSM to the above problem yields the following algorithm:

$$v^{k+1} = J_{\mathcal{A}}^{\beta}(2J_{\mathcal{B}}^{\beta} - I)v^k + (I - J_{\mathcal{B}}^{\beta})v^k, \tag{4.30}$$

$$\lambda^{k+1} = J_{\mathcal{B}}^{\beta} v^{k+1}, \tag{4.31}$$

where $J_{\mathcal{A}}^{\beta} = (I + \beta\mathcal{A})^{-1}$ and $J_{\mathcal{B}}^{\beta} = (I + \beta\mathcal{B})^{-1}$ are the resolvent operators. After some calculation, it can be shown that this algorithm is equivalent to the classic ADMM (Algorithm 1) [20]. Here, the variable v corresponds to

$$v^k = \beta A x^k + \lambda^k. \tag{4.32}$$

The linear convergence of DRSM was established by Lions and Merciers [40]. We summarize their result in the following theorem.

Theorem 4.4 (Lions and Mercier [40])**.** *Assume the operator $\mathcal{B}$ is both coercive and Lipschitz. Namely, there exists $\alpha > 0$ and $M > 0$ such that*

$$\langle \mathcal{B}(\lambda_1) - \mathcal{B}(\lambda_2),\ \lambda_1 - \lambda_2 \rangle \geq \alpha \|\lambda_1 - \lambda_2\|^2, \tag{4.33}$$

$$\|\mathcal{B}(\lambda_1) - \mathcal{B}(\lambda_2)\| \leq M \|\lambda_1 - \lambda_2\|. \tag{4.34}$$

Then, there exists a constant $C > 0$ such that

$$\|\lambda^k - \lambda^*\|^2 \leq C \cdot \theta^k,\ \|v^{k+1} - v^*\|^2 \leq \theta \cdot \|v^k - v^*\|^2, \tag{4.35}$$

where

$$\theta = 1 - \frac{2\beta\alpha}{(1 + \beta M)^2}. \tag{4.36}$$

The smallest θ is given by

$$\theta_{\min} = 1 - \frac{\alpha}{2M}, \tag{4.37}$$

which corresponds to $\beta = 1/M$.

Under the assumptions of Scenario 1 of Table 4.1, f is strongly convex with constant ν_f, gradient ∇f is Lipschitz with constant L_f, and matrix A has full row rank. Then the operator $\mathcal{B} := \partial[f^* \circ A^\top]$ is coercive and Lipschitz with the constants

$$\alpha = \lambda_{\min}(AA^\top)/L_f,\ M = \|A\|^2/\nu_f. \tag{4.38}$$

Hence, the linear convergence of ADMM follows from Theorem 4.4, and the optimal linear rate is given by

$$\theta_{\min} = 1 - \frac{\lambda_{\min}(AA^\top)\nu_f}{2\|A\|^2 L_f} = 1 - \frac{1}{2\kappa_A^2 \kappa_f}. \tag{4.39}$$

In contrast, our linear rate (4.19) is given by

$$\frac{1}{1+\delta_{\max}} = 1 - \delta_{\max} + O(\delta_{\max}^2) = 1 - \frac{1}{\kappa_A\sqrt{\kappa_f}} + O\left(\frac{1}{\kappa_A^2 \kappa_f}\right), \tag{4.40}$$

which is better than (4.39). By careful inspection, it is also clear that our linear rate (4.18) considerably improves the classic rate (4.36) in [40].

4.3 Applications

This section describes several well-known optimization models on which Algorithm 2 not only enjoys global linear convergence but also often has easy-to-solve subproblems.

4.3.1 Convex Regularization

Consider the convex regularization models discussed in Section 2.3:

$$\min_x f(Ax - b) + g(x), \tag{4.41}$$

where f is the loss function and g is the regularization function. In many applications, the loss function f is typically a *strongly convex* function with Lipschitz continuous gradient, such as the commonly used squared ℓ_2-norm $\|\cdot\|_2^2$ for least squares. If matrix A has full column rank, then the term $f(Ax-b)$ is strongly convex in x and the following reformulation of the problem (4.41):

$$\begin{aligned} &\min_{x,y} \ f(Ax - b) + g(y) \\ &\text{s.t. } x - y = 0, \end{aligned} \tag{4.42}$$

satisfies the scenario 1 in Table 4.1.

If matrix A is column-rank deficient, then $f(Ax-b)$ is not strongly convex in x. Alternatively, we may consider the following equivalent problem after introducing $y = Ax - b$:

$$\begin{aligned} &\min_{x,y} \ g(x) + f(y) \\ &\text{s.t. } Ax - y = b. \end{aligned} \tag{4.43}$$

Then it satisfies the scenario 1 in Table 4.1 if the x-subproblem can be solved exactly with $P = 0$ (since full column rank of A is needed if $P \succ 0$). Moreover, if the regularization function g is also strongly convex, then the problem will satisfy the scenorio 2 or 4, depending on whether g has Lipschitz continuous gradient. As one of the most commonly used regularization methods, the *Tikhonov regularization*, also known as *ridge regression* in statistics:

$$\min_x \|Ax-b\|^2 + \lambda\|x\|^2 \ (\lambda > 0) \tag{4.44}$$

is an example of scenario 4, that both f and g are strongly convex and have Lipschitz continuous gradients.

4.3.2 Sparse Optimization

Elastic net (augmented ℓ_1) model:

$$\min_x \ \|x\|_1 + \alpha\|x\|^2 + \frac{1}{2\mu}\|Ax-b\|^2, \tag{4.45}$$

where $A \in \mathbb{R}^{m\times n}$, $\alpha > 0$ and $\mu > 0$ are parameters. It has been shown that the elastic model can effectively recover sparse vectors and outperform Lasso ($\alpha = 0$) on reported real-world regression problems [63]. With the constraint $x = y$, (4.45) can

be reformulated as:

$$\begin{aligned} &\min_{x,y} \ \|y\|_1 + \alpha\|x\|^2 + \frac{1}{2\mu}\|Ax - b\|^2 \\ &\text{s.t. } x - y = 0. \end{aligned} \tag{4.46}$$

Similarly, the elastic net model can be extended for recovering low-rank matrices.

Augmented nuclear-norm model:

$$\min_{X} \ \|X\|_* + \alpha\|X\|_F^2 + \frac{1}{2\mu}\|\mathcal{A}(X) - b\|^2, \tag{4.47}$$

where $\alpha > 0$ and $\mu > 0$ are parameters, $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ is a linear operator, $\|\cdot\|_F$ denotes the Frobenius norm, and $\|\cdot\|_*$ denotes the nuclear norm. By variable splitting $X = Y$, (4.47) can be reformulated as:

$$\begin{aligned} &\min_{X,Y} \ \|Y\|_* + \alpha\|X\|_F^2 + \frac{1}{2\mu}\|\mathcal{A}(X) - b\|^2 \\ &\text{s.t. } X - Y = 0. \end{aligned} \tag{4.48}$$

In (4.46) and (4.48), both of the functions $f(x) = \alpha\|x\|^2 + \frac{1}{2\mu}\|Ax - b\|^2$ and $f(X) = \alpha\|X\|_F^2 + \frac{1}{2\mu}\|\mathcal{A}(X) - b\|^2$ are strongly convex and have Lipschitz continuous gradients. Therefore, they satisfy the scenario 1 of Table 4.1.

4.3.3 Consensus and Sharing Optimization

As discussed in Section 2.3, we consider the global consensus problem with regularization:

$$\begin{aligned} &\min_{\{x_i\},y} \ \sum_{i=1}^{N} f_i(x_i) + g(y) \\ &\text{s.t. } \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} - \begin{bmatrix} -\mathbf{I} \\ \vdots \\ -\mathbf{I} \end{bmatrix} y = 0, \end{aligned} \tag{4.49}$$

and the sharing problem:

$$\min_{\{x_i\},\{y_i\}} \sum_{i=1}^{N} f_i(x_i) + g\left(\sum_{i=1}^{N} y_i\right), \quad \text{s.t. } x_i - y_i = 0,\ i = 1, \ldots, N, \tag{4.50}$$

where f_i's are local cost functions and g is the regularization or shared cost function.

If each function f_i is strongly convex and has Lipschitz continuous gradient, then both problems (4.49) and (4.50) satisfy the conditions in scenario 1 of Table 4.1. If the function g is strongly convex and both f and g have Lipschitz continuous gradients, then problem (4.49) satisfies scenario 3 or 4, depending on whether each function f_i is strongly convex.

4.4 Numerical Demonstration

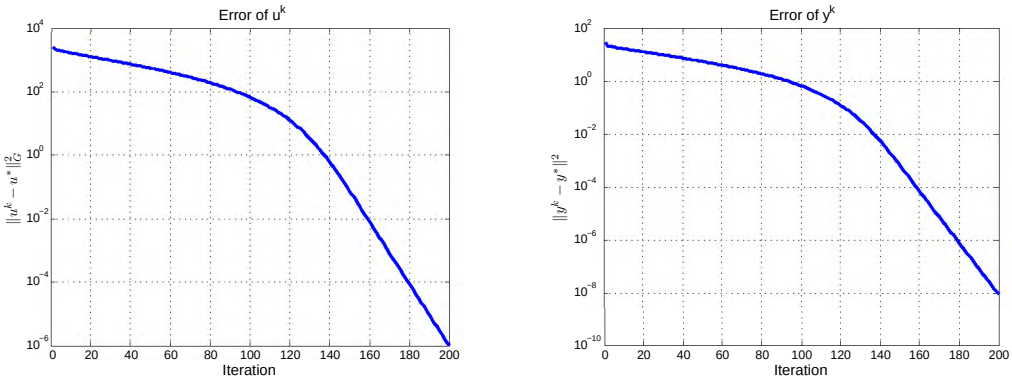
We present some simple numerical results to demonstrate the linear convergence of Algorithm 2 on the *elastic net* and *distributed Lasso* problems.

4.4.1 Elastic Net

We apply Algorithm 2 with $P = 0$ and $Q = 0$ to a small elastic net problem (4.46), where the feature matrix A has $m = 250$ examples and $n = 1000$ features. We first generated the matrix A from the standard Gaussian distribution $\mathcal{N}(0, 1)$ and then orthonormalized its rows. A sparse vector $x^0 \in \mathbb{R}^n$ was generated with 25 nonzero entries, each sampled from the standard Gaussian distribution. The observation vector $b \in \mathbb{R}^m$ was then computed by $b = Ax^0 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 10^{-3}I)$. We chose the model parameters $\alpha = 0.1$ and $\mu = 10^{-2}$, which we found to yield reasonable accuracy for recovering the sparse solution. We initialized all the variables at zero and set the algorithm parameters $\beta = 100$ and $\gamma = 1$. We ran the algorithm for 200 iterations and recorded the errors at each iteration with respect to a precomputed

reference solution u^*.

Figure 4.1(a) shows the decreasing behavior of $\|u^k - u^*\|_G^2 (:= \beta\|x^k - x^*\|^2 + \|\lambda^k - \lambda^*\|^2/\beta)$ as the algorithm progresses. Since variable y is not contained in the G-norm, we also plot the convergence curve of $\|y^k - y^*\|^2$ in Figure 4.1(b). We observe that both u^k and y^k converge at similar linear rates. In addition, the convergence appears to have different stages. The later stage exhibits faster convergence rate than the earlier stage. This can be clearly seen in Figure 4.2 which depicts the Q-linear rate $\|u^{k+1} - u^*\|_G^2/\|u^k - u^*\|_G^2$.



(a) $\|u^k - u^*\|_G^2$ vs. iteration

(b) $\|y^k - y^*\|^2$ vs. iteration

Figure 4.1 : Elastic net: convergence curves of ADMM.

Here, the strong convexity constant of f is $\nu_f = 2\alpha + \lambda_{\min}(A^TA)/\mu = 2\alpha$ and the Lipschitz constant of ∇f is $L_f = 2\alpha + \lambda_{\max}(A^TA)/\mu = 2\alpha + 1/\mu$. By (4.18), our bound for the global linear rate is $(1+\delta)^{-1} = 0.996$, which roughly matches the early-stage rate shown in the figure. However, our theoretical bound is rather conservative, since it is a global worst-case bound and it does not take into account the properties of the ℓ_1 norm and the solution. In fact, the optimal solution x^* is very sparse and x^k will also become sparse after a number of iterations. Let $\mathcal{S}$ be an

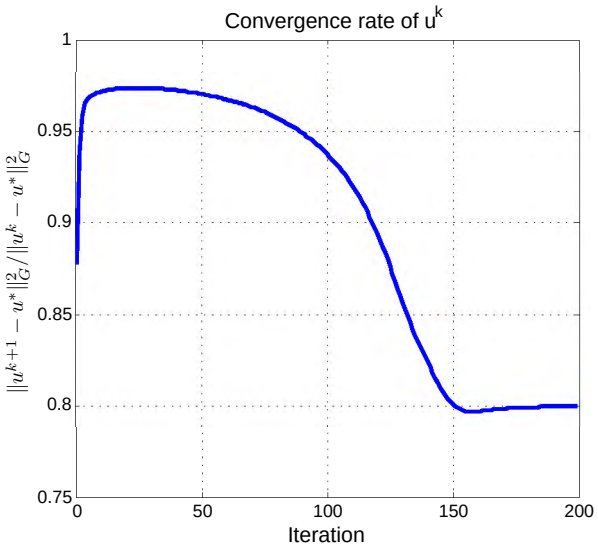


Figure 4.2 : Elastic net: Q-linear convergence rate of ADMM.

index set of the nonzero support of $(x^k - x^*)$, and $A_{\mathcal{S}}$ be a submatrix composed of those columns of A indexed by $\mathcal{S}$. Then, the constants ν_f and L_f in our bound can be effectively replaced by $\bar{\nu}_f = 2\alpha + \lambda_{\min}(A_{\mathcal{S}}^T A_{\mathcal{S}})/\mu$ and $\bar{L}_f = 2\alpha + \lambda_{\max}(A_{\mathcal{S}}^T A_{\mathcal{S}})/\mu$, thereby accounting for the faster convergence rate in the later stage. For example, letting $\mathcal{S}$ be the nonzero support of the optimal solution x^*, we obtain an estimate of the (asymptotic) linear rate $(1+\delta)^{-1} = 0.817$, which well matches the later-stage rate.

4.4.2 Distributed Lasso

We consider solving the Lasso problem in a distributed way [41]:

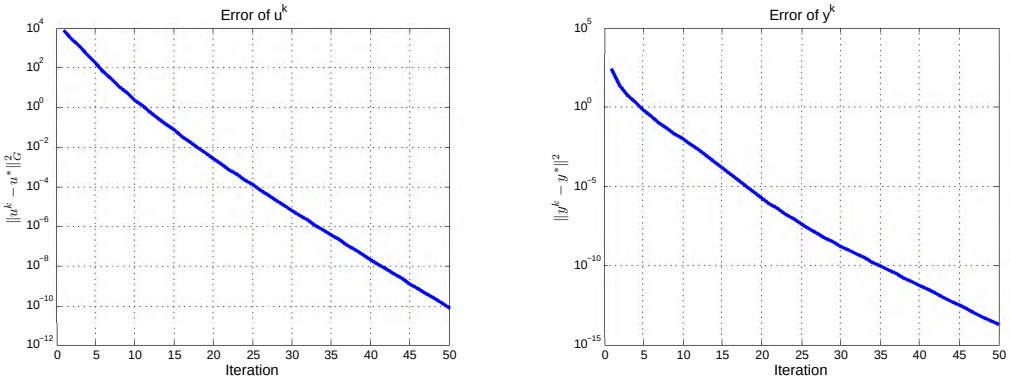
$$\begin{aligned} \min_{\{x_i\},y} \quad & \sum_{i=1}^{N} \frac{1}{2\mu}\|A_i x_i - b_i\|^2 + \|y\|_1 \\ \text{s.t.} \quad & x_i - y = 0,\ i = 1,\ldots,N, \end{aligned} \tag{4.51}$$

which is an instance of the global consensus problem with regularization (2.21).

We apply Algorithm 2 with $P = 0$ and $Q = 0$ to a small distributed Lasso problem (4.51) with $N = 5$, where each A_i has $m = 600$ examples and $n = 500$ features. Each A_i is a tall matrix and has full column rank, yielding a strongly convex objective function in x_i. Therefore, Algorithm 2 is guaranteed to converge linearly.

We generated the data similarly as in the elastic net test. We randomly generated each A_i from the standard Gaussian distribution $\mathcal{N}(0, 1)$, and then simply scaled its columns to have a unit length. We generated a sparse vector $x^0 \in \mathbb{R}^n$ with 250 nonzero entries, each sampled from the $\mathcal{N}(0, 1)$ distribution. Each $b_i \in \mathbb{R}^m$ was then computed by $b_i = A_i x^0 + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, 10^{-3} I)$. We chose the model parameter $\mu = 0.1$, which we found to yield reasonably good recovery quality. From the initial point at zero, we ran the algorithm with parameters $\beta = 10$ and $\gamma = 1$ for 50 iterations and computed the iterative errors.

Figure 4.3 demonstrates the clear linear convergence behavior of $\|u^k - u^*\|_G^2$ and $\|y^k - y^*\|^2$. In Figure 4.4, the Q-linear convergence rate of $\|u^k - u^*\|_G^2$ is depicted. For this problem, the strong convexity constant is $\nu_f = \min_i\{\lambda_{\min}(A_i^T A_i)/\mu\}$ and the Lipschitz constant is $L_f = \max_i\{\lambda_{\max}(A_i^T A_i)/\mu\}$. However, the condition number ν_f/L_f in this test is relatively big, and hence the theoretical linear rate specified by (4.19) is not a very tight bound for the observed fast rate. Note that all x_i's tend to be equal and become sparse after a number of iterations. Similar to our previous discussion in Section 4.4.1, we can estimate the asymptotic linear rate by letting $\bar{\nu}_f = \lambda_{\min}(A_{\mathcal{S}}^T A_{\mathcal{S}})/(\mu N)$ and $\bar{L}_f = \lambda_{\max}(A_{\mathcal{S}}^T A_{\mathcal{S}})/(\mu N)$, where $A \in \mathbb{R}^{Nm \times n}$ is formed by stacking all the matrices A_i $(i = 1, \ldots, N)$, and $\mathcal{S}$ is an index set of the nonzero support of x^*. We obtained the asymptotic linear rate to be $(1 + \delta)^{-1} = 0.779$, which appears to be a much tighter bound.



(a) $\|u^k - u^*\|_G^2$ vs. iteration (b) $\|y^k - y^*\|^2$ vs. iteration

Figure 4.3 : Distributed Lasso: convergence curves of ADMM.

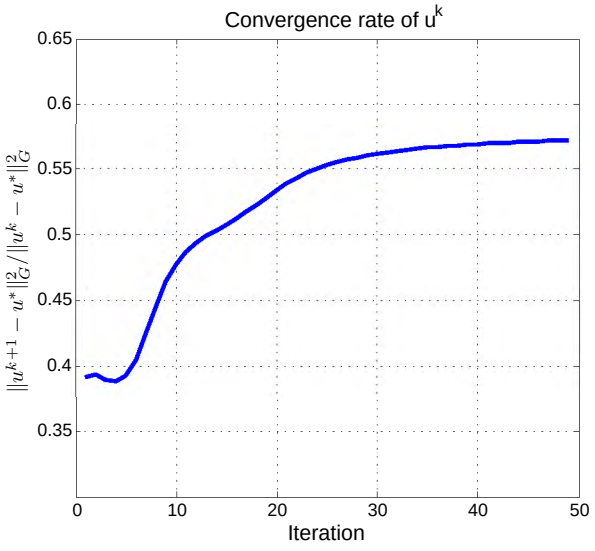


Figure 4.4 : Distributed Lasso: Q-linear convergence rate of ADMM.

Chapter 5

Parallel Multi-Block ADMM

This chapter introduces a parallel and multi-block extension to the alternating direction method of multipliers (ADMM) for solving convex problem:

$$\begin{aligned} \text{minimize} \quad & f_1(\mathbf{x}_1) + \cdots + f_N(\mathbf{x}_N) \\ \text{subject to} \quad & A_1\mathbf{x}_1 \ + \cdots + A_N\mathbf{x}_N = c, \\ & \mathbf{x}_1 \in \mathcal{X}_1, \ \ \ldots, \ \mathbf{x}_N \in \mathcal{X}_N. \end{aligned}$$

The proposed algorithm decomposes the original problem into N smaller subproblems and solves them in parallel at each iteration. It is well suited to distributed computing and is particularly attractive for solving certain large-scale problems.

This chapter is organized as follows. In Section 5.1, we briefly review some existing parallel and distributed algorithms for solving the above optimization problem. Then we introduce a few novel results in the following sections. In Section 5.2, we show that extending ADMM straightforwardly from the classic Gauss-Seidel setting to the Jacobi setting, from 2 blocks to N blocks, will preserve convergence if matrices A_i are mutually near-orthogonal and have full column-rank. In Section 5.3, for general matrices A_i, we propose to add proximal terms of different kinds to the N subproblems so that the subproblems can be solved in flexible and efficient ways and the algorithm converges globally at a rate of $o(1/k)$. We also introduce a strategy for dynamically tuning the parameters in the algorithm, often leading to substantial acceleration of the convergence in practice. In Section 5.4, numerical results are presented to demonstrate

the efficiency of the proposed algorithm in comparison with several existing parallel algorithms. We also implemented our algorithm on Amazon EC2, an on-demand public computing cloud, and report its performance on very large-scale basis pursuit problems with distributed data.

5.1 Introduction

We consider the following convex optimization problem with N ($N \geq 2$) blocks of variables:

$$\min_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N} \sum_{i=1}^{N} f_i(\mathbf{x}_i) \quad \text{s.t.} \sum_{i=1}^{N} A_i\mathbf{x}_i = c, \tag{5.1}$$

where $\mathbf{x}_i \in \mathbb{R}^{n_i}$, $A_i \in \mathbb{R}^{m\times n_i}$, $c \in \mathbb{R}^m$, and $f_i : \mathbb{R}^{n_i} \to (-\infty, +\infty]$ are closed proper convex functions, $i = 1, 2, \ldots, N$. If an individual block is subject to constraint $\mathbf{x}_i \in \mathcal{X}_i$, where $\mathcal{X}_i \subseteq \mathbb{R}^{n_i}$ is a nonempty closed convex set, it can be incorporated in the objective function f_i using the indicator function:

$$I_{\mathcal{X}_i}(\mathbf{x}_i) = \begin{cases} 0 & \text{if } \mathbf{x}_i \in \mathcal{X}_i, \\ +\infty & \text{otherwise.} \end{cases} \tag{5.2}$$

In this study, we do not impose any additional assumptions such as strict convexity or differentiability on the objective functions f_i.

The problem (5.1) is also referred to as an *extended monotropic programming problem* [2]. The special case that each $\mathbf{x}_i$ is a scalar (i.e., $n_i = 1$) is called a *monotropic programming problem* [51]. Such optimization problems arise from a broad spectrum of applications including numerical partial differential equations, signal and image processing, compressive sensing, statistics and machine learning. See [1, 4, 8, 22, 43, 46, 47, 55, 60] and the references therein for a number of examples.

We focus on *parallel and distributed* optimization algorithms for solving the problem (5.1). Due to the dramatically increasing demand for dealing with big data,

parallel and distributed computational methods are highly desirable. Since both of the objective function and constraints of (5.1) are summations of terms on individual $\mathbf{x}_i$'s (we call them *separable*), the problem can be decomposed into N smaller subproblems, which can be solved in a parallel and distributed manner.

5.1.1 Literature review

A simple distributed algorithm for solving (5.1) is *dual decomposition* [19], which is essentially a *dual ascent method* or *dual subgradient method* [53] as follows. Consider the Lagrangian for problem (5.1):

$$\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) - \lambda^\top \left(\sum_{i=1}^{N} A_i \mathbf{x}_i - c \right) \tag{5.3}$$

where $\lambda \in \mathbb{R}^m$ is the Lagrangian multiplier or the dual variable. The method of dual decomposition iterates as follows: for $k \geq 1$,

$$\begin{cases} (\mathbf{x}_1^{k+1}, \mathbf{x}_2^{k+1}, \ldots, \mathbf{x}_N^{k+1}) = \arg\min_{\{\mathbf{x}_i\}} \ \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda^k), \\ \lambda^{k+1} = \lambda^k - \alpha_k \left(\sum_{i=1}^{N} A_i \mathbf{x}_i^{k+1} - c \right), \end{cases} \tag{5.4}$$

where $\alpha_k > 0$ is a step-size. Since all the $\mathbf{x}_i$'s are separable in the Lagrangian function (5.3), the $\mathbf{x}$-update step reduces to solving N individual $\mathbf{x}_i$-subproblems:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) - \langle \lambda^k, A_i \mathbf{x}_i \rangle, \ \text{for } i = 1, 2, \ldots, N, \tag{5.5}$$

and thus they can be carried out in parallel. With suitable choice of α_k and certain assumptions, dual decomposition is guaranteed to converge to an optimal solution [53]. However, the convergence of such subgradient method often tends to be slow in practice. Its convergence rate for general convex problems is $O(1/\sqrt{k})$.

Another effective distributed approach is based on the *alternating direction method of multipliers* (ADMM). As we know, ADMM was introduced to solve the special case

of problem (5.1) with only two blocks of variables (i.e., $N = 2$). To solve the problem (5.1) with $N \geq 3$, one can first convert the multi-block problem into an equivalent two-block problem and then solve it by ADMM. The problem reformulation is done via variable splitting [1]:

$$\begin{aligned} \min_{\{\mathbf{x}_i\},\{\mathbf{z}_i\}} \quad & \sum_{i=1}^{N} f_i(\mathbf{x}_i) \\ \text{s.t.} \quad & A_i\mathbf{x}_i - \mathbf{z}_i = \frac{c}{N}, \ \forall i = 1, 2, \ldots, N, \\ & \sum_{i=1}^{N} \mathbf{z}_i = 0, \end{aligned} \tag{5.6}$$

or equivalently,

$$\begin{aligned} \min_{\{\mathbf{x}_i\},\{\mathbf{z}_i\}} \quad & \sum_{i=1}^{N} f_i(\mathbf{x}_i) + I_{\mathcal{Z}}(\mathbf{z}_1, \ldots, \mathbf{z}_N) \\ \text{s.t.} \quad & A_i\mathbf{x}_i - \mathbf{z}_i = \frac{c}{N}, \ \forall i = 1, 2, \ldots, N, \end{aligned} \tag{5.7}$$

where $I_{\mathcal{Z}}$ is an indicator function defined by (5.2), and the convex set $\mathcal{Z}$ is given by

$$\mathcal{Z} = \left\{ (\mathbf{z}_1, \ldots, \mathbf{z}_N) : \sum_{i=1}^{N} \mathbf{z}_i = 0 \right\}.$$

The variables $\mathbf{z}_i$ (we call them *splitting variables*) are introduced to decouple the $\mathbf{x}_i$'s in the constraints. Now we group the variables into two blocks: $\mathbf{x} := (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ and $\mathbf{z} := (\mathbf{z}_1, \ldots, \mathbf{z}_N)$. Then ADMM can be applied directly. The augmented Lagrangian for (5.7) is given by

$$\mathcal{L}_\beta(\mathbf{x}, \mathbf{z}, \lambda) = \sum_{i=1}^{N} f_i(\mathbf{x}_i) + I_{\mathcal{Z}}(\mathbf{z}) - \sum_{i=1}^{N} \lambda_i^\top \left(A_i\mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} \right) + \frac{\beta}{2} \sum_{i=1}^{N} \left\| A_i\mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} \right\|^2. \tag{5.8}$$

Since all the $\mathbf{x}_i$'s are now fully decoupled, the resulting $\mathbf{x}$-subproblem decomposes into N individual $\mathbf{x}_i$-subproblems:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2} \left\| A_i\mathbf{x}_i - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\beta} \right\|^2, \tag{5.9}$$

which can be carried out in parallel. The resulting $\mathbf{z}$-subproblem is a simple quadratic problem:

$$\mathbf{z}^{k+1} = \underset{\{\mathbf{z}:\sum_{i=1}^N \mathbf{z}_i=0\}}{\arg\min} \sum_{i=1}^N \frac{\beta}{2} \left\| A_i\mathbf{x}_i - \mathbf{z}_i - \frac{c}{N} - \frac{\lambda_i^k}{\beta} \right\|^2, \tag{5.10}$$

which admits a closed-form solution:

$$\mathbf{z}_i^{k+1} = \left(A_i\mathbf{x}_i^k - \frac{c}{N} - \frac{\lambda_i^k}{\beta} \right) - \frac{1}{N} \left\{ \sum_{j=1}^N A_j\mathbf{x}_j^k - \frac{c}{N} - \frac{\lambda_j^k}{\beta} \right\}. \tag{5.11}$$

We summarize the algorithm below:

Algorithm 3: Variable Splitting ADMM (VSADMM)

Initialize $\mathbf{x}^0,\ \lambda^0,\ \beta > 0$;

for $k = 0,\ 1, \ldots$ **do**

  Update $\mathbf{z}_i$ then $\mathbf{x}_i$ for $i = 1, \ldots, N$ *in parallel* by:

  $\mathbf{z}_i^{k+1} = \left(A_i\mathbf{x}_i^k - \frac{c}{N} - \frac{\lambda_i^k}{\beta} \right) - \frac{1}{N} \left\{ \sum_{j=1}^N A_j\mathbf{x}_j^k - \frac{c}{N} - \frac{\lambda_j^k}{\beta} \right\}$;

  $\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2} \left\| A_i\mathbf{x}_i - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\beta} \right\|^2$;

  Update $\lambda_i^{k+1} = \lambda_i^k - \beta \left(A_i\mathbf{x}_i^{k+1} - \mathbf{z}_i^{k+1} - \frac{c}{N} \right), \forall i = 1, \ldots, N.$

This ADMM approach based on (5.7), by introducing splitting variables, *substantially increases the number of variables and constraints* in the problem, especially when N is large. Alternatively, we aim to develop a *parallel* and *multi-block* extension of ADMM that solves (5.1) directly without splitting variables.

A straightforward multi-block extension of ADMM is to use a sweep of Gauss-Seidel update to minimize the augmented Lagrangian. Namely, it updates $\mathbf{x}_i$ for

$i = 1, 2, \ldots, N$ sequentially as follows:

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \ \mathcal{L}_\beta(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{i-1}^{k+1}, \mathbf{x}_i, \mathbf{x}_{i+1}^{k}, \ldots, \mathbf{x}_N^{k}, \lambda^k) \\ &= \arg\min_{\mathbf{x}_i} \ f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\| \sum_{j<i} A_j \mathbf{x}_j^{k+1} + A_i \mathbf{x}_i + \sum_{j>i} A_j \mathbf{x}_j^{k} - c - \frac{\lambda^k}{\beta} \right\|^2, \end{aligned} \tag{5.12}$$

where $\mathcal{L}_\beta$ is the augmented Lagrangian for (5.1):

$$\mathcal{L}_\beta(\mathbf{x}_1, \ldots, \mathbf{x}_N, \lambda) = \sum_{i=1}^N f_i(\mathbf{x}_i) - \lambda^\top \left(\sum_{i=1}^N A_i \mathbf{x}_i - c \right) + \frac{\beta}{2} \left\| \sum_{i=1}^N A_i \mathbf{x}_i - c \right\|^2. \tag{5.13}$$

Such *Gauss-Seidel ADMM* (Algorithm 4) has been considered lately, e.g., in [34, 38]. It has been shown that the algorithm may not converge for $N \geq 3$ [9]. Although lack of convergence guarantee, some empirical studies show that the Gauss-Seidel ADMM is still very effective at solving many practical problems (see, e.g., [47, 55, 58]). However, as the Gauss-Seidel ADMM updates the blocks sequentially one after another, it is not amenable for parallelization.

Algorithm 4: Gauss-Seidel ADMM

1 Initialize $\mathbf{x}^0,\ \lambda^0,\ \beta > 0$;

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

3 Update $\mathbf{x}_i$ for $i = 1, \ldots, N$ *sequentially* by:
$\mathbf{x}_i^{k+1} = \min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\| \sum_{j<i} A_j \mathbf{x}_j^{k+1} + A_i \mathbf{x}_i + \sum_{j>i} A_j \mathbf{x}_j^{k} - c - \frac{\lambda^k}{\beta} \right\|^2$;

4 Update $\lambda^{k+1} = \lambda^k - \beta \left(\sum_{i=1}^N A_i \mathbf{x}_i^{k+1} - c \right)$.

5.1.2 Jacobi-Type ADMM

A parallel counterpart of the Gauss-Seidel ADMM is the Jacobi ADMM (see Algorithm 5 below). Unlike the Gauss-Seidel ADMM, the Jacobi ADMM updates all the

blocks in parallel at every iteration:

$$\begin{aligned}\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \ \mathcal{L}_\beta(\mathbf{x}_1^k, \ldots, \mathbf{x}_{i-1}^k, \mathbf{x}_i, \mathbf{x}_{i+1}^k, \ldots, \mathbf{x}_N^k, \lambda^k) \\ &= \arg\min_{\mathbf{x}_i} \ f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\| A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2, \ \forall i = 1, \ldots, N. \end{aligned} \tag{5.14}$$

Therefore, it is well suited to parallel and distributed computation.

Algorithm 5: Jacobi ADMM

1 Initialize $\mathbf{x}^0,\ \lambda^0,\ \beta > 0$;

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

3 Update $\mathbf{x}_i$ for $i = 1, \ldots, N$ *in parallel* by:
$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\| A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2$;

4 Update $\lambda^{k+1} = \lambda^k - \beta\left(\sum_{i=1}^N A_i\mathbf{x}_i^{k+1} - c\right)$.

On the other hand, such straightforward parallelization comes with a cost: the augmented Lagrangian is minimized "less accurately" than it is in the Gauss-Seidel scheme. Hence, this Jacobi ADMM is more likely to diverge than the Gauss-Seidel ADMM. In fact, it may diverge even when $N = 2$; see [33] for such an example. A few variants of Jacobi ADMM have been proposed which take certain correction steps to ensure its convergence [32, 33].

In order to guarantee the convergence of Jacobi ADMM, either additional assumptions or modifications to Algorithm 5 must be made. Therefore, we discuss along these two lines in the rest of the chapter. In Section 5.2, we provide a sufficient condition for the convergence of Algorithm 5 by assuming the "near-orthogonality" of the matrices A_i. In Section 5.3, we propose a simple modification to Algorithm 5 — called Jacobi-Proximal ADMM — which converges at a rate of $o(1/k)$.

5.1.3 Notation and Assumptions

To simplify the notation, we introduce

$$\mathbf{x} := \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{pmatrix} \in \mathbb{R}^n, \ A := \begin{pmatrix} A_1, \ldots, A_N \end{pmatrix} \in \mathbb{R}^{m \times n}, \ \mathbf{u} := \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \in \mathbb{R}^{n+m},$$

where

$$n = \sum_{i=1}^{N} n_i.$$

Throughout this chapter, we make the following standard assumptions.

Assumption 5.1. *Functions* $f_i : \mathbb{R}^{n_i} \to (-\infty, +\infty]$ $(i = 1, 2, \ldots, N)$ *are closed proper convex.*

Assumption 5.2. *There exists a saddle point* $\mathbf{u}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_N^*, \lambda^*)$ *to the problem* (5.1)*. Namely,* $\mathbf{u}^*$ *satisfies the KKT conditions:*

$$A_i^\top \lambda^* \in \partial f_i(\mathbf{x}_i^*), \ \textit{for } i = 1, \ldots, N, \tag{5.15}$$

$$A\mathbf{x}^* = \sum_{i=1}^{N} A_i \mathbf{x}_i^* = c. \tag{5.16}$$

The optimality conditions (5.15) *and* (5.16) *can be written in a more compact form by the following variational inequality [33]:*

$$f(\mathbf{x}) - f(\mathbf{x}^*) + (\mathbf{u} - \mathbf{u}^*)^\top F(\mathbf{u}^*) \geq 0, \ \forall u, \tag{5.17}$$

where $f(\mathbf{x}) := \sum_i f_i(\mathbf{x}_i)$ *and*

$$F(\mathbf{u}) := \begin{pmatrix} -A_1^\top \lambda \\ \vdots \\ -A_N^\top \lambda \\ A\mathbf{x} - c \end{pmatrix}.$$

5.2 On the Convergence of Jacobi ADMM

In this section, we provide a sufficient condition to guarantee the convergence of Jacobi ADMM (Algorithm 5). The condition only depends on the coefficient matrices A_i, without imposing further assumptions on the objective functions f_i or the penalty parameter β. For the Gauss-Seidel ADMM (Algorithm 4), a sufficient condition for convergence is given in [9] for the special case $N = 3$, assuming two of the three coefficient matrices are orthogonal. Our condition does not require exact orthogonality. Instead, we mainly assume that the matrices $A_i,\ i = 1, 2, \ldots, N$ are mutually "near-orthogonal" and have full column-rank.

Theorem 5.1. *Suppose that there exists* $\delta >= 0$ *such that*

$$\|A_i^\top A_j\| \leq \delta,\ \forall\ i \neq j,\ \text{and}\ \lambda_{min}(A_i^\top A_i) > 3(N-1)\delta,\ \forall\ i, \tag{5.18}$$

where $\lambda_{min}(A_i^\top A_i)$ *denotes the smallest eigenvalue of* $A_i^\top A_i$. *Then the sequence* $\{\mathbf{u}^k\}$ *generated by Algorithm 5 converges to a solution* $\mathbf{u}^*$ *to the problem (5.1).*

The proof technique is motivated by the contraction analysis of the sequence $\{\mathbf{u}^k\}$ under some G-norm (e.g., see [13, 33, 36]). To prove the theorem, we first need the following lemma:

Lemma 5.1. *Let*

$$G_0 := \begin{pmatrix} \beta A_1^\top A_1 & & & \\ & \ddots & & \\ & & \beta A_N^\top A_N & \\ & & & \frac{1}{\beta}\mathbf{I} \end{pmatrix},\ S_0 := \begin{pmatrix} \beta A_1^\top A_1 & & & A_1^\top \\ & \ddots & & \vdots \\ & & \beta A_N^\top A_N & A_N^\top \\ A_1 & \ldots & A_N & \frac{1}{\beta}\mathbf{I} \end{pmatrix},$$

where $\mathbf{I}$ *is the identity matrix of size* $m \times m$. *For* $k \geq 1$, *we have*

$$\|\mathbf{u}^k - \mathbf{u}^*\|_{G_0}^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_{G_0}^2 \geq \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{S_0}^2, \tag{5.19}$$

where

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{S_0}^2 = \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{G_0}^2 + 2(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}). \tag{5.20}$$

This lemma follows directly from Lemma 5.2 (which will be proved in the next section), since it is a special case with $\gamma = 1$ and $P_i = 0,\ \forall i$. Now we are ready to prove the theorem.

Proof of Theorem 5.1. By the assumption $\|A_i^\top A_j\| \leq \delta,\ i \neq j$, we have

$$\left|\sum_{i\neq j}\langle A_i\mathbf{a}_i, A_j\mathbf{b}_j\rangle\right| \leq \sum_{i\neq j}\delta\|\mathbf{a}_i\|\|\mathbf{b}_j\| \leq \frac{\delta}{2}(N-1)(\|\mathbf{a}\|^2 + \|\mathbf{b}\|^2),\ \forall\ \mathbf{a}, \mathbf{b}. \tag{5.21}$$

To simplify the notation, we let

$$\mathbf{a}_i^k := \mathbf{x}_i^k - \mathbf{x}_i^*,\ i = 1, 2, \ldots, N. \tag{5.22}$$

Note that

$$\lambda^k - \lambda^{k+1} = \beta A\mathbf{a}^{k+1},\ \mathbf{x}^k - \mathbf{x}^{k+1} = \mathbf{a}^k - \mathbf{a}^{k+1}.$$

Then, we can rewrite (5.20) as

$$\frac{1}{\beta}\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_{S_0}^2 = \sum_i \|A_i(\mathbf{a}_i^k - \mathbf{a}_i^{k+1})\|^2 + \|A\mathbf{a}^{k+1}\|^2 + 2\langle A\mathbf{a}^{k+1},\ A(\mathbf{a}^k - \mathbf{a}^{k+1})\rangle \tag{5.23}$$

$$= \sum_i \|A_i\mathbf{a}_i^k\|^2 + 2\sum_{i\neq j}\langle A_i\mathbf{a}_i^{k+1}, A_j\mathbf{a}_j^k\rangle - \sum_{i\neq j}\langle A_i\mathbf{a}_i^{k+1}, A_j\mathbf{a}_j^{k+1}\rangle \tag{5.24}$$

$$\geq \sum_i \|A_i\mathbf{a}_i^k\|^2 - (N-1)\delta(\|\mathbf{a}^{k+1}\|^2 + \|\mathbf{a}^k\|^2) - (N-1)\delta\|\mathbf{a}^{k+1}\|^2 \tag{5.25}$$

$$= \sum_i \|A_i\mathbf{a}_i^k\|^2 - (N-1)\delta\|\mathbf{a}^k\|^2 - 2(N-1)\delta\|\mathbf{a}^{k+1}\|^2, \tag{5.26}$$

where the inequality (5.25) comes from (5.21). By Lemma 5.1, we have

$$\begin{aligned}&\|\mathbf{u}^k-\mathbf{u}^*\|_{G_0}^2-2(N-1)\delta\beta\|\mathbf{a}^k\|^2\\ \geq&\|\mathbf{u}^{k+1}-\mathbf{u}^*\|_{G_0}^2-2(N-1)\delta\beta\|\mathbf{a}^{k+1}\|^2+\beta\sum_i\|A_i\mathbf{a}_i^k\|^2-3(N-1)\delta\beta\|\mathbf{a}^k\|^2.\end{aligned}\tag{5.27}$$

We further simplify (5.27) as

$$b^k-b^{k+1}\geq d^k,\tag{5.28}$$

where the sequences $\{b^k\}$ and $\{d^k\}$ are defined by

$$b^k:=\|\mathbf{u}^k-\mathbf{u}^*\|_{G_0}^2-2(N-1)\delta\beta\|\mathbf{a}^k\|^2,\tag{5.29}$$

$$d^k:=\beta\sum_i\|A_i\mathbf{a}_i^k\|^2-3(N-1)\delta\beta\|\mathbf{a}^k\|^2.\tag{5.30}$$

By the definition of G_0, we have

$$b^k=\beta\sum_i\|A_i\mathbf{a}_i^k\|^2-2(N-1)\delta\beta\|\mathbf{a}_i^k\|^2+\frac{1}{\beta}\|\lambda^k-\lambda^*\|^2.\tag{5.31}$$

Since we assume $\lambda_{\min}(A_i^\top A_i)>3(N-1)\delta$, it follows that

$$\|A_i\mathbf{a}_i^k\|^2\geq 3(N-1)\delta\|\mathbf{a}_i^k\|^2,\ \forall i.\tag{5.32}$$

Then it is easy to see that $b^k\geq 0$ and $d^k\geq 0$. By (5.28), the nonnegative sequence $\{b^k\}$ is monotonically non-increasing. Hence, $\{b^k\}$ converges to some $b^*\geq 0$. By (5.28), it also follows that $d^k\to 0$. Therefore, $\mathbf{a}^k\to 0$, i.e., $\mathbf{x}^k\to\mathbf{x}^*$.

Next we show $\lambda^k\to\lambda^*$. By taking limit of (5.31) and using $\mathbf{a}^k\to 0$, we have

$$b^*=\lim_{k\to\infty}b^k=\lim_{k\to\infty}\frac{1}{\beta}\|\lambda^k-\lambda^*\|^2.\tag{5.33}$$

To show $\lambda^k\to\lambda^*$, it thus suffices to show $b^*=0$.

By (5.33), $\{\lambda^k\}$ is bounded and must have a convergent subsequence $\lambda^{k_j}\to\bar{\lambda}$. Recall the optimality conditions for the $\mathbf{x}_i$-subproblems (5.14):

$$A_i^\top\left(\lambda^k-\beta(A_i\mathbf{x}_i^{k+1}+\sum_{j\neq i}A_j\mathbf{x}_j^k-c)\right)\in\partial f_i(\mathbf{x}_i^{k+1}).\tag{5.34}$$

By Theorem 24.4 of [52], taking limit over the subsequence $\{k_j\}$ on both sides of (5.34) yields:

$$A_i^\top \bar{\lambda} \in \partial f_i(\mathbf{x}_i^*),\ \forall\ i. \tag{5.35}$$

Therefore, $(\mathbf{x}^*, \bar{\lambda})$ satisfies the KKT conditions of the problem (5.1). Since $(\mathbf{x}^*, \lambda^*)$ is any KKT point, now we let $\lambda^* = \bar{\lambda}$. By (5.33) and $\|\lambda^{k_j} - \lambda^*\|^2 \to 0$, we must have $b^* = 0$, thereby completing the proof. □

5.3 Jacobi-Proximal ADMM

In this section, we propose the *Jacobi-Proximal ADMM* (Algorithm 6). Compared with Algorithm 5, we introduce a proximal term $\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2$ ($P_i \succeq 0$) for each $\mathbf{x}_i$-subproblem and a damping parameter $\gamma > 0$ for the update of λ.

Algorithm 6: Jacobi-Proximal ADMM

1 Initialize: $\mathbf{x}_i^0$ $(i = 1, 2, \ldots, N)$ and λ^0;

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

3 Update $\mathbf{x}_i$ for $i = 1, \ldots, N$ *in parallel* by:
$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2 + \frac{1}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|_{P_i}^2$;

4 Update $\lambda^{k+1} = \lambda^k - \gamma\beta(\sum_{i=1}^N A_i\mathbf{x}_i^{k+1} - c)$.

The proposed algorithm has a few advantages. First of all, as we will show, it enjoys global convergence as well as an $o(1/k)$ convergence rate under conditions on P_i and γ. Secondly, when the $\mathbf{x}_i$-subproblem is not strictly convex, adding the proximal term can make the subproblem strictly or strongly convex, making it more stable. Thirdly, we provide multiple choices for matrices P_i with which the subproblems can

be made easier to solve. For example, in many cases that $A_i^\top A_i$ is ill-conditioned or computationally expensive to invert, the $\mathbf{x}_i$-subproblem becomes difficult to solve exactly which contains a quadratic term $\frac{\beta}{2}\mathbf{x}_i A_i^\top A_i \mathbf{x}_i$. One can let $P_i = D_i - \beta A_i^\top A_i$, which effectively replaces the quadratic term $\frac{\beta}{2}\mathbf{x}_i A_i^\top A_i \mathbf{x}_i$ by $\frac{1}{2}\mathbf{x}_i D_i \mathbf{x}_i$. The matrix D_i can be chosen as some well-conditioned and simple matrix (such as identity matrix, diagonal matrix and so on), thereby leading to an easier subproblem. Various choices of P_i have been discussed in Section 3.2.1. The use of flexible proximal terms makes it possible to solve its subproblems in different ways, important for easy coding and fast computation.

In this section, we mainly study the convergence of the Jacobi-Proximal ADMM. We first show its convergence and then establish an $o(1/k)$ convergence rate in the same sense as in [35]. Furthermore, we discuss how to tune the parameters in order to make the algorithm more practical. Our numerical results on the exchange problem and the basis pursuit problem show that the proposed algorithm achieves competitive performance, in comparison with several existing parallel algorithms.

5.3.1 Convergence

To simplify the notation, we let

$$G_x := \begin{pmatrix} P_1 + \beta A_1^\top A_1 & & \\ & \ddots & \\ & & P_N + \beta A_N^\top A_N \end{pmatrix}, \; G := \begin{pmatrix} G_x & \\ & \frac{1}{\gamma\beta}\mathbf{I} \end{pmatrix},$$

and

$$S := \begin{pmatrix} P_1 + \beta A_1^\top A_1 & & & \frac{1}{\gamma}A_1^\top \\ & \ddots & & \vdots \\ & & P_N + \beta A_N^\top A_N & \frac{1}{\gamma}A_N^\top \\ \frac{1}{\gamma}A_1 & \dots & \frac{1}{\gamma}A_N & \frac{2-\gamma}{\beta\gamma^2}\mathbf{I} \end{pmatrix}, \tag{5.36}$$

where $\mathbf{I}$ is the identity matrix of size $m \times m$. In the rest of the section, we let $\{\mathbf{u}^k\}$ denote the sequence generated by Jacobi-Proximal ADMM from any initial point. The analysis is based on bounding the error $\|\mathbf{u}^k - \mathbf{u}^*\|_G^2$ and estimating its decrease, motivated by the works [13, 33, 36].

Lemma 5.2. *For $k \geq 1$, we have*

$$\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_G^2 \geq \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_S^2, \tag{5.37}$$

where

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_S^2 = \|\mathbf{x}^k - \mathbf{x}^{k+1}\|_{G_x}^2 + \frac{2-\gamma}{\beta\gamma^2}\|\lambda^k - \lambda^{k+1}\|^2 + \frac{2}{\gamma}(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}). \tag{5.38}$$

Proof. Recall that in Algorithm 6, we solve the following $\mathbf{x}_i$-subproblem:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} f_i(\mathbf{x}_i) + \frac{\beta}{2}\left\| A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2 + \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2.$$

Its optimality condition is given by

$$A_i^\top\left(\lambda^k - \beta(A_i\mathbf{x}_i^{k+1} + \sum_{j\neq i} A_j\mathbf{x}_j^k - c)\right) + P_i(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}) \in \partial f_i(\mathbf{x}_i^{k+1}). \tag{5.39}$$

For convenience, we introduce $\hat{\lambda} := \lambda^k - \beta(A\mathbf{x}^{k+1} - c)$. Then (5.39) can be rewritten as

$$A_i^\top\left(\hat{\lambda} - \beta\sum_{j\neq i} A_j(\mathbf{x}_j^k - \mathbf{x}_j^{k+1})\right) + P_i(\mathbf{x}_i^k - \mathbf{x}_i^{k+1}) \in \partial f_i(\mathbf{x}_i^{k+1}). \tag{5.40}$$

By Lemma 2.1, it follows from (5.15) and (5.40) that

$$\left\langle A_i(\mathbf{x}_i^{k+1}-\mathbf{x}_i^*),\ \hat{\lambda}-\lambda^*-\beta\sum_{j\neq i}A_j(\mathbf{x}_j^k-\mathbf{x}_j^{k+1})\right\rangle+(\mathbf{x}_i^{k+1}-\mathbf{x}_i^*)^\top P_i(\mathbf{x}_i^k-\mathbf{x}_i^{k+1})\geq 0.$$

Summing the above inequality over all i and using the following equality for each i:

$$\sum_{j\neq i}A_j(\mathbf{x}_j^k-\mathbf{x}_j^{k+1})=A(\mathbf{x}^k-\mathbf{x}^{k+1})-A_i(\mathbf{x}_i^k-\mathbf{x}_i^{k+1}),$$

we obtain

$$\begin{aligned}&\langle A(\mathbf{x}^{k+1}-\mathbf{x}^*),\hat{\lambda}-\lambda^*\rangle+\sum_{i=1}^N(\mathbf{x}_i^{k+1}-\mathbf{x}_i^*)^\top(P_i+\beta A_i^\top A_i)(\mathbf{x}_i^k-\mathbf{x}_i^{k+1})\\&\geq\beta\langle A(\mathbf{x}^{k+1}-\mathbf{x}^*),A(\mathbf{x}^k-\mathbf{x}^{k+1})\rangle.\end{aligned}\tag{5.41}$$

Note that

$$A(\mathbf{x}^{k+1}-\mathbf{x}^*)=\frac{1}{\gamma\beta}(\lambda^k-\lambda^{k+1}),$$

and

$$\hat{\lambda}-\lambda^*=(\hat{\lambda}-\lambda^{k+1})+(\lambda^{k+1}-\lambda^*)=\frac{\gamma-1}{\gamma}(\lambda^k-\lambda^{k+1})+(\lambda^{k+1}-\lambda^*).$$

With the above two equations, the inequality (5.41) can be rewritten as

$$\begin{aligned}&\langle\frac{1}{\gamma\beta}(\lambda^k-\lambda^{k+1}),\lambda^{k+1}-\lambda^*\rangle+\sum_{i=1}^N(\mathbf{x}_i^{k+1}-\mathbf{x}_i^*)^\top(P_i+\beta A_i^\top A_i)(\mathbf{x}_i^k-\mathbf{x}_i^{k+1})\\&\geq\frac{1-\gamma}{\gamma^2\beta}\|\lambda^k-\lambda^{k+1}\|^2+\frac{1}{\gamma}(\lambda^k-\lambda^{k+1})^\top A(\mathbf{x}^k-\mathbf{x}^{k+1}),\end{aligned}\tag{5.42}$$

or more compactly,

$$(\mathbf{u}^k-\mathbf{u}^{k+1})^\top G(\mathbf{u}^{k+1}-\mathbf{u}^*)\geq\frac{1-\gamma}{\gamma^2\beta}\|\lambda^k-\lambda^{k+1}\|^2+\frac{1}{\gamma}(\lambda^k-\lambda^{k+1})^\top A(\mathbf{x}^k-\mathbf{x}^{k+1}).\tag{5.43}$$

Since $\|\mathbf{u}^k-\mathbf{u}^*\|_G^2-\|\mathbf{u}^{k+1}-\mathbf{u}^*\|_G^2=2(\mathbf{u}^k-\mathbf{u}^{k+1})^\top G(\mathbf{u}^{k+1}-\mathbf{u}^*)+\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_G^2$, using the above inequality (5.43) yields (5.37) immediately. □

If the matrix S (5.36) is positive definite, there exists some $\eta > 0$ such that

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_S^2 \geq \eta \cdot \|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2 \geq 0. \tag{5.44}$$

Then Lemma 5.2 indicates that

$$\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 - \|\mathbf{u}^{k+1} - \mathbf{u}^*\|_G^2 \geq \eta \cdot \|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2. \tag{5.45}$$

That is, the iterative sequence $\{\mathbf{u}^k\}$ is *strictly contractive*. In particular, the error $\|\mathbf{u}^k - \mathbf{u}^*\|_G^2$ is monotonically non-increasing and thus converging, as well as $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2 \to 0$. Then the convergence of the algorithm ($\|\mathbf{u}^k - \mathbf{u}^*\|_G^2 \to 0$) follows immediately from the standard analysis for contraction methods (see, e.g., [31]). We omit the details of the proof for the sake of brevity and state the convergence theorem as follows.

Theorem 5.2. *Suppose the parameters in Algorithm 6 satisfy that the matrix S (5.36) is positive definite. Then the sequence $\{\mathbf{u}^k\}$ generated by Algorithm 6 converges to a solution $\mathbf{u}^*$ to the problem (5.1).*

In the following theorem, we give a sufficient condition to guarantee that S is positive definite. It basically requires the parameters P_i $(i = 1, 2, \ldots, N)$ to be sufficiently large.

Theorem 5.3. *Suppose the parameters β, γ and P_i $(i = 1, 2, \ldots, N)$ satisfy the following condition:*

$$\begin{cases} P_i \succ \beta(\frac{1}{\epsilon_i} - 1)A_i^\top A_i, \ i = 1, 2, \ldots, N \\ \sum_{i=1}^N \epsilon_i < 2 - \gamma, \end{cases} \tag{5.46}$$

for some $\epsilon_i > 0$, $i = 1, 2, \ldots, N$. Then the matrix S (5.36) is positive definite. Thus, the sequence $\{\mathbf{u}^k\}$ generated by Algorithm 6 converges to a solution $\mathbf{u}^*$ to the problem (5.1).

The condition (5.46) *can be reduced to*

$$P_i \succ \beta\left(\frac{N}{2-\gamma}-1\right)A_i^\top A_i,\ \ i=1,2,\ldots,N, \tag{5.47}$$

by letting each $\epsilon_i < \frac{2-\gamma}{N}$. *In particular, for the following choices:*

- $P_i = \tau_i \mathbf{I}$ *(standard proximal), condition* (5.47) *becomes* $\tau_i > \beta\left(\frac{N}{2-\gamma}-1\right)\|A_i\|^2$;
- $P_i = \tau_i \mathbf{I} - \beta A_i^\top A_i$ *(prox-linear), condition* (5.47) *becomes* $\tau_i > \frac{\beta N}{2-\gamma}\|A_i\|^2$.

Proof. For any $\mathbf{u} = (\mathbf{x};\lambda) \in \mathbb{R}^{n+m}$, we have

$$\|\mathbf{u}\|_S^2 := \|\mathbf{x}\|_{G_x}^2 + \frac{2-\gamma}{\beta\gamma^2}\|\lambda\|^2 + \frac{2}{\gamma}\lambda^\top A\mathbf{x}. \tag{5.48}$$

Using the following basic inequality:

$$\frac{2}{\gamma}\lambda^\top A\mathbf{x} = \sum_{i=1}^N \frac{2}{\gamma}\lambda^\top A_i\mathbf{x}_i \geq -\sum_{i=1}^N\left(\frac{\epsilon_i}{\beta\gamma^2}\|\lambda\|^2 + \frac{\beta}{\epsilon_i}\|A_i\mathbf{x}_i\|^2\right), \tag{5.49}$$

for any $\epsilon_i > 0$ $(i = 1,2,\ldots,N)$, we have

$$\|\mathbf{u}\|_S^2 \geq \sum_{i=1}^N \|\mathbf{x}_i\|^2_{P_i+\beta A_i^\top A_i - \frac{\beta}{\epsilon_i}A_i^\top A_i} + \frac{2-\gamma-\sum_{i=1}^N\epsilon_i}{\beta\gamma^2}\|\lambda\|^2. \tag{5.50}$$

The condition (5.46) guarantees that $P_i+\beta A_i^\top A_i - \frac{\beta}{\epsilon_i}A_i^\top A_i \succ 0$ and $2-\gamma-\sum_{i=1}^N\epsilon_i > 0$, and thus $\|\mathbf{u}\|_S > 0$. Therefore, S is positive definite. The rest follows immediately. □

Under the similar near-orthogonality assumption on the matrices A_i, we have the following convergence result for Jacobi-Proximal ADMM:

Theorem 5.4. *Suppose there exists* $\delta \geq 0$ *such that* $\|A_i^\top A_j\| \leq \delta$ *for all* $i \neq j$, *and the parameters in Algorithm 6 satisfy the following condition: for some* $\alpha, \rho > 0$ *and* $0 < \gamma < 2$,

$$\begin{cases} P_i \succ \beta(\frac{1}{\alpha}-1)A_i^\top A_i + \frac{\beta}{\rho}\delta(N-1)\mathbf{I} \\ \lambda_{\min}(A_i^\top A_i) > \frac{2-\gamma+\rho}{2-\gamma-\alpha}\delta(N-1) \end{cases} \text{for } i=1,\ldots,N. \tag{5.51}$$

Then Algorithm 6 converges to a solution to the problem (5.1).

Proof. Let

$$H := \begin{pmatrix} A_1^\top A_1 & & \\ & \ddots & \\ & & A_N^\top A_N \end{pmatrix}.$$

If $\|A_i^\top A_j\| \le \delta$ for all $i \neq j$, then it is easy to show the following: for any $\mathbf{x}$ and $\mathbf{y}$,

$$\begin{aligned} \|A\mathbf{x}\|^2 &= \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 + \sum_{i\neq j} \mathbf{x}_i^\top A_i^\top A_j \mathbf{x}_j \ge \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 - \delta \sum_{i\neq j} \|\mathbf{x}_i\|\|\mathbf{x}_j\| \\ &\ge \sum_{i=1}^N \|A_i\mathbf{x}_i\|^2 - \delta(N-1)\|\mathbf{x}\|^2 = \|\mathbf{x}\|^2_{[H-\delta(N-1)\mathbf{I}]}, \end{aligned} \tag{5.52}$$

and

$$\begin{aligned} 2\mathbf{x}^\top A^\top A\mathbf{y} &= 2\sum_{i=1}^N \mathbf{x}_i^\top A_i^\top A_j \mathbf{y}_j + 2\sum_{i\neq j} \mathbf{x}_i^\top A_i^\top A_j \mathbf{y}_j \qquad (5.53) \\ &\ge 2\sum_{i=1}^N \mathbf{x}_i^\top A_i^\top A_j \mathbf{y}_j - 2\delta \sum_{i\neq j} \|\mathbf{x}_i\|\|\mathbf{y}_j\| \\ &\ge -\sum_{i=1}^N \alpha\|A_i\mathbf{x}_i\|^2 - \rho\delta(N-1)\|\mathbf{x}\|^2 - \sum_{i=1}^N \frac{1}{\alpha}\|A_i\mathbf{y}_i\|^2 - \frac{1}{\rho}\delta(N-1)\|\mathbf{y}\|^2 \\ &= -\|\mathbf{x}\|^2_{[\alpha H+\rho\delta(N-1)\mathbf{I}]} - \|\mathbf{y}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\rho}\delta(N-1)\mathbf{I}]},\ \forall \alpha,\rho>0, \qquad (5.54) \end{aligned}$$

Using the above inequalities, we have

$$\begin{aligned} &\frac{2}{\gamma}(\lambda^k - \lambda^{k+1})^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}) = 2\beta(\mathbf{x}^{k+1} - \mathbf{x}^*)A^\top A(\mathbf{x}^k - \mathbf{x}^{k+1}) \\ \ge& -\beta\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2_{[\alpha H+\rho\delta(N-1)\mathbf{I}]} - \beta\|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\rho}\delta(N-1)\mathbf{I}]}, \end{aligned} \tag{5.55}$$

and

$$\|\lambda^k - \lambda^{k+1}\|^2 = \gamma^2\beta^2\|A(\mathbf{x}^{k+1} - \mathbf{x}^*)\|^2 \ge \gamma^2\beta^2\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2_{[H-\delta(N-1)\mathbf{I}]}. \tag{5.56}$$

Therefore,

$$\begin{aligned} \|\mathbf{u}^k - \mathbf{u}^{k+1}\|_S^2 \ge& \|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2_{G_x} + (2-\gamma)\beta\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2_{[H-\delta(N-1)\mathbf{I}]} \\ &- \beta\|\mathbf{x}^{k+1} - \mathbf{x}^*\|^2_{[\alpha H+\rho\delta(N-1)\mathbf{I}]} - \beta\|\mathbf{x}^k - \mathbf{x}^{k+1}\|^2_{[\frac{1}{\alpha}H+\frac{1}{\rho}\delta(N-1)\mathbf{I}]}. \end{aligned} \tag{5.57}$$

As long as the following holds:

$$\begin{cases} G_x \succ \frac{\beta}{\alpha}H + \frac{\beta}{\rho}\delta(N-1)\mathbf{I}, \\ (2-\gamma)\beta[H - \delta(N-1)\mathbf{I}] \succ \beta[\alpha H + \rho\delta(N-1)\mathbf{I}], \end{cases} \tag{5.58}$$

which is equivalent to the condition (5.51), there must exist some $\eta > 0$ such that (5.44) and (5.45) hold. Then the convergence of Algorithm 6 follows immediately from the standard analysis of contraction methods [31]. □

Remark 5.1. *The conditions in Theorem 5.4 guarantee that (5.44) holds, i.e., there exists some $\eta > 0$ such that*

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_S^2 \geq \ \eta \cdot \|\mathbf{u}^k - \mathbf{u}^{k+1}\|^2.$$

But the matrix S is not necessarily positive semi-definite. Also, the term $\dfrac{2-\gamma+\rho}{2-\gamma-\alpha}$ in (5.51) may be negative for some α. Then, $\lambda_{min}(A_i^\top A_i)$ is allowed to be 0; in other words, A_i may not be of full column rank. As long as the conditions in (5.51) are satisfied, Algorithm 6 will converge.

5.3.2 Convergence Rate of $o(1/k)$

Next, we shall establish the $o(1/k)$ convergence rate of Jacobi-Proximal ADMM. We use the quantity $\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2$ as a measure of the convergence rate motivated by [33, 35]. Here, we define the matrix M by

$$M := \begin{pmatrix} M_x & \\ & \frac{1}{\gamma\beta}\mathbf{I} \end{pmatrix} \quad \text{and} \quad M_x := G_x - \beta A^\top A.$$

Theorem 5.5. *If $S \succ 0$ and $M_x \succeq 0$, then*

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2 = o(1/k),$$

and, thus,

$$\|\mathbf{x}^k - \mathbf{x}^{k+1}\|_{M_x}^2 = o(1/k) \quad \textit{and} \quad \|\lambda^k - \lambda^{k+1}\|^2 = o(1/k).$$

We need the following monotonic property of the iterations:

Lemma 5.3. *If $M_x \succeq 0$ and $0 < \gamma < 2$, then*

$$\|\mathbf{u}^k - \mathbf{u}^{k+1}\|_M^2 \leq \|\mathbf{u}^{k-1} - \mathbf{u}^k\|_M^2. \tag{5.59}$$

Proof. Let $\Delta\mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \mathbf{x}_i^{k+1}, i = 1, \ldots, N$, $\Delta\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{x}^{k+1}$, and $\Delta\lambda^{k+1} = \lambda^k - \lambda^{k+1}$. By Lemma 2.1, the optimality conditions (5.40) at k-th and $(k+1)$-th iterations yield

$$\langle A_i\Delta\mathbf{x}_i^{k+1}, \Delta\lambda^k - \beta A\Delta\mathbf{x}^{k+1} - \beta\sum_{j\neq i} A_j(\Delta\mathbf{x}_j^k - \Delta\mathbf{x}_j^{k+1})\rangle + (\Delta\mathbf{x}_i^{k+1})^\top P_i(\Delta\mathbf{x}_i^k - \Delta\mathbf{x}_i^{k+1}) \geq 0. \tag{5.60}$$

Summing up over all i and rearranging the terms, we have

$$\langle A\Delta\mathbf{x}^{k+1}, \Delta\lambda^k\rangle \geq \|\Delta\mathbf{x}^{k+1}\|_{G_x}^2 - (\Delta\mathbf{x}^k)^\top(G_x - \beta A^\top A)\Delta\mathbf{x}^{k+1}. \tag{5.61}$$

Since $M_x := G_x - \beta A^\top A \succeq 0$, we have

$$2(\Delta\mathbf{x}^k)^\top(G_x - \beta A^\top A)\Delta\mathbf{x}^{k+1} \leq \|\Delta\mathbf{x}^k\|_{M_x}^2 + \|\Delta\mathbf{x}^k\|_{M_x}^2, \tag{5.62}$$

and thus

$$\begin{aligned} 2\langle A\Delta\mathbf{x}^{k+1}, \Delta\lambda^k\rangle &\geq \|\Delta\mathbf{x}^{k+1}\|_{2G_x - M_x}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2 \\ &= \|\Delta\mathbf{x}^{k+1}\|_{G_x + \beta A^\top A}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2. \end{aligned} \tag{5.63}$$

Note that $\Delta\lambda^{k+1} = \Delta\lambda^k - \gamma\beta A\Delta\mathbf{x}^{k+1}$. It follows that

$$\begin{aligned} \frac{1}{\gamma\beta}\|\Delta\lambda^k\|^2 - \frac{1}{\gamma\beta}\|\Delta\lambda^{k+1}\|^2 &= 2\langle A\Delta\mathbf{x}^{k+1}, \Delta\lambda^k\rangle - \gamma\beta\|A\Delta\mathbf{x}^{k+1}\|^2 \\ &\geq \|\Delta\mathbf{x}^{k+1}\|_{G_x + (1-\gamma)\beta A^\top A}^2 - \|\Delta\mathbf{x}^k\|_{M_x}^2, \end{aligned} \tag{5.64}$$

or equivalently,

$$\begin{aligned}&(\|\Delta\mathbf{x}^k\|_{M_x}^2+\frac{1}{\gamma\beta}\|\Delta\lambda^k\|^2)-(\|\Delta\mathbf{x}^{k+1}\|_{M_x}+\frac{1}{\gamma\beta}\|\Delta\lambda^{k+1}\|^2)\\&\geq\|\Delta\mathbf{x}^{k+1}\|_{(2-\gamma)\beta A^\top A}^2\geq 0,\end{aligned}\tag{5.65}$$

which completes the proof. □

Proof of Theorem 5.5. By Lemma 5.2 and $S\succ 0$, there must exist $\eta>0$ such that

$$\|\mathbf{u}^k-\mathbf{u}^*\|_G^2-\|\mathbf{u}^{k+1}-\mathbf{u}^*\|_G^2\geq\eta\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_G^2\geq\eta\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_M^2.\tag{5.66}$$

Summing (5.66) over k gives

$$\sum_{k=1}^{\infty}\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_M^2<\infty.\tag{5.67}$$

On the other hand, Lemma 5.3 implies the monotone non-increasing of $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_M^2$. By Lemma 3.2, we have $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_M^2=o(1/k)$, which completes the proof. □

5.3.3 Adaptive Parameter Tuning

The parameters satisfying the condition (5.46) may be rather conservative, because the inequality (5.49) for bounding $\|\mathbf{u}\|_S^2$ is usually very loose. In practice, we can compute $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_S^2$ exactly at very little extra cost. If $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_S^2>0$, then Lemma 5.2 assures the decreasing of the solution error (in the G-norm) so that the current parameters are acceptable. On the other hand, if $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_S^2<0$, then the matrix S is not positive definite, suggesting that the current parameters P_i, $i=1,2,\ldots,N$ may be too small. So we should make $\{P_i\}$ bigger until $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_S^2>0$ holds. Therefore, we propose a practical strategy for adaptively adjusting the parameters $\{P_i\}$ based on the value of $\|\mathbf{u}^k-\mathbf{u}^{k+1}\|_S^2$:

Initialize with small $P_i^0 \succeq 0$ $(i = 1, 2, \ldots, N)$ and a small $\eta > 0$;
for $k = 1, 2, \ldots$ **do**
  if $\|\mathbf{u}^{k-1} - \mathbf{u}^k\|_S^2 > \eta \cdot \|\mathbf{u}^{k-1} - \mathbf{u}^k\|^2$ **then**
    $P_i^{k+1} \leftarrow P_i^k, \ \forall i$;
  else
    Increase P_i: $P_i^{k+1} \leftarrow \alpha_i P_i^k + \beta_i Q_i$ $(\alpha_i > 1, \ \beta_i \geq 0, \ Q_i \succ 0), \forall i$;
    Restart: $\mathbf{u}^k \leftarrow \mathbf{u}^{k-1}$;

The above strategy starts with relatively small proximal parameters $\{P_i\}$ and gradually increase them. By Theorem 5.3, we know that when the parameters $\{P_i\}$ are large enough for (5.46) to hold, the condition (5.44) will be satisfied (for sufficiently small η). Therefore, the adjustment of $\{P_i\}$ cannot occur infinite times. After a finite number of iterations, $\{P_i\}$ will remain constant and the contraction property (5.45) of the iterations will hold. Therefore, the convergence of such an adaptive parameter tuning scheme follows immediately from our previous analysis.

Theorem 5.6. *Suppose the matrices P_i $(i = 1, 2, \ldots, N)$ in Algorithm 6 are adaptively adjusted using the above scheme. Then the algorithm converges to a solution to the problem (5.1).*

Empirical evidence shows that the parameters $\{P_i\}$ typically adjust themselves only during the first few iterations and then remain constant afterwards. Alternatively, one may also decrease the parameters after every few iterations or after they have not been updated for a certain number of iterations. But the total times of decrease should be bounded to guarantee convergence. By using this adaptive strategy, the resulting parameters $\{P_i\}$ are usually much smaller than those required by the

condition (5.46), thereby leading to substantially faster convergence in practice.

5.4 Numerical Experiments

In this section, we present numerical results to compare the performance of the following parallel splitting algorithms:

- **Prox-JADMM**: proposed Jacobi-Proximal ADMM (Algorithm 6);
- **VSADMM**: Variable Splitting ADMM (Algorithm 3);
- **Corr-JADMM**: Jacobi ADMM with correction steps [33]. At every iteration, it first generates a "predictor" $\tilde{\mathbf{u}}^{k+1}$ by an iteration of Jacobi ADMM (Algorithm 5) and then corrects $\tilde{\mathbf{u}}^{k+1}$ to generate the new iterate by:

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \alpha_k(\mathbf{u}^k - \tilde{\mathbf{u}}^{k+1}), \tag{5.68}$$

  where $\alpha_k > 0$ is a step size. In our experiments, we adopt the dynamically updated step size α_k according to [33], which is shown to converge significantly faster than using a constant step size, though updating the step size requires extra computation.

- **YALL1**: one of the state-of-the-art solvers for the ℓ_1-minimization problem.

In Section 5.4.1 and 5.4.2, all of the numerical experiments are run in MATLAB (R2011b) on a workstation with an Intel Core i5-3570 CPUs (3.40GHz) and 32 GB of RAM. Section 5.4.3 gives two very large instances that are solved by a C/MPI implementation on Amazon Elastic Compute Cloud (EC2).

5.4.1 Exchange Problem

Consider a network of N agents that exchange n commodities. Let $\mathbf{x}_i \in \mathbb{R}^n$ $(i = 1, 2, \ldots, N)$ denote the amount of commodities that are exchanged among the N agents. Each agent i has a certain cost function $f_i : \mathbb{R}^n \to \mathbb{R}$. The exchange problem (see, e.g., [4] for a review) is given by

$$\min_{\{\mathbf{x}_i\}} \sum_{i=1}^{N} f_i(\mathbf{x}_i) \quad \text{s.t.} \sum_{i=1}^{N} \mathbf{x}_i = 0, \tag{5.69}$$

which minimizes the total cost among N agents subject to an equilibrium constraint on the commodities. This is a special case of (5.1) where $A_i = \mathbf{I}$ and $c = 0$.

We consider quadratic cost functions $f_i(\mathbf{x}_i) := \frac{1}{2}\|C_i\mathbf{x}_i - d_i\|^2$, where $C_i \in \mathbb{R}^{p\times n}$ and $d_i \in \mathbb{R}^p$. Then all the compared algorithms solve the following type of subproblems at every iteration:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \ \frac{1}{2}\|C_i\mathbf{x}_i - d_i\|^2 + \frac{\beta}{2}\|\mathbf{x}_i - b_i^k\|^2, \ \forall i = 1, 2, \ldots, N, \tag{5.70}$$

except that Prox-JADMM also adds a proximal term $\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_i^k\|_{P_i}^2$. Here $b_i^k \in \mathbb{R}^m$ is a vector independent of $\mathbf{x}_i$ and takes different forms in different algorithms. For Prox-JADMM, we simply set $P_i = \tau_i\mathbf{I}$ $(\tau_i > 0)$. Clearly, each $\mathbf{x}_i$-subproblem is a quadratic program that can be computed efficiently using various methods.

In our experiment, we randomly generate $\mathbf{x}_i^*$, $i = 1, 2, \ldots, N-1$, following the standard Gaussian distribution, and let $\mathbf{x}_N^* = -\sum_{i=1}^{N-1}\mathbf{x}_i^*$. Matrices C_i are random Gaussian matrices, and vectors d_i are computed by $d_i = C_i\mathbf{x}_i^*$. Apparently, $\mathbf{x}^*$ is a solution (not necessarily unique) to (5.69), and the optimal objective value is 0.

The penalty parameter β is set to be 0.01, 1 and 0.01 for Prox-JADMM, VSADMM and Corr-JADMM, respectively. They are nearly optimal for each algorithm, picked out of a number of different values. Note that the parameter for VSADMM is quite

different from the other two algorithms because it has different constraints due to the variable splitting. For Prox-JADMM, the proximal parameters are initialized by $\tau_i = 0.1(N-1)\beta$ and adaptively updated by the strategy in Subsection 5.3.3; the parameter γ is set to be 1.

The size of the test problem is set to be $n = 100,\ N = 100,\ p = 80$. Letting all the algorithms run 200 iterations, we plot their objective value $\sum_{i=1}^{N} f_i(\mathbf{x}_i)$ and residual $\|\sum_{i=1}^{N} \mathbf{x}_i\|_2$. Note that the per-iteration cost (in terms of both computation and communication) is roughly the same for all the compared algorithms. Figure 5.1 shows the comparison result, which is averaged over 100 random trials. We can see that Prox-JADMM is clearly the fastest one among the compared algorithm.

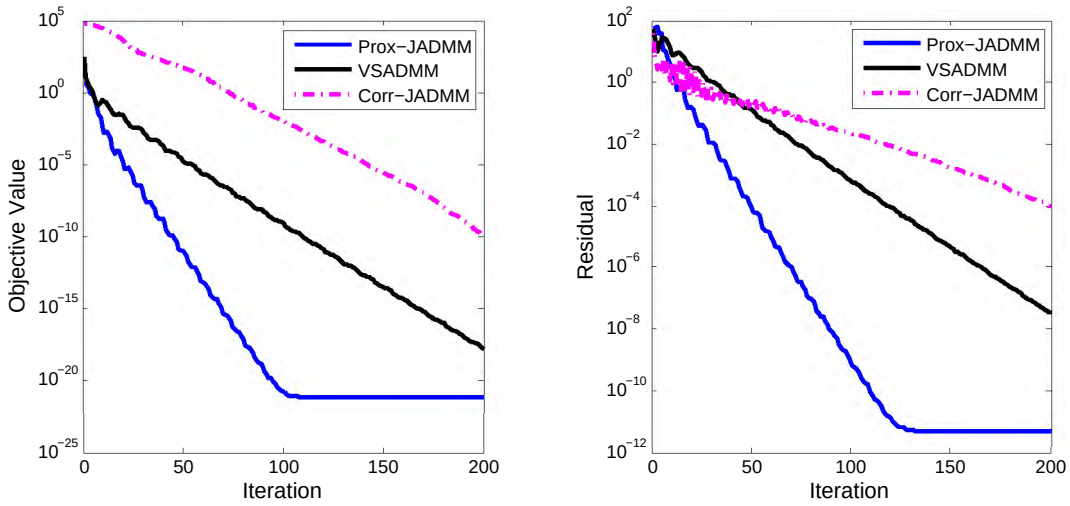


Figure 5.1 : Exchange problem ($n = 100,\ N = 100,\ p = 80$).

5.4.2 Basis Pursuit

We consider the ℓ_1-minimization problem for finding sparse solutions of an underdetermined linear system:

$$\min_{\mathbf{x}} \ \|\mathbf{x}\|_1 \quad \text{s.t. } A\mathbf{x} = c, \tag{5.71}$$

where $\mathbf{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $c \in \mathbb{R}^m$ $(m < n)$. It is also known as the *basis pursuit* problem, which has been widely used in compressive sensing, signal and image processing, statistics, and machine learning. Suppose that the data is partitioned into N blocks: $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ and $A = [A_1, A_2, \ldots, A_N]$. Then the problem (5.71) can be written in the form of (5.1) with $f_i(\mathbf{x}_i) = \|\mathbf{x}_i\|_1$.

In our experiment, a sparse solution $\mathbf{x}^*$ is randomly generated with k $(k \ll n)$ nonzeros drawn from the standard Gaussian distribution. Matrix A is also randomly generated from the standard Gaussian distribution, and it is partitioned evenly into N blocks. The vector c is then computed by $c = A\mathbf{x}^* + \eta$, where $\eta \sim \mathcal{N}(0,\ \sigma^2\mathbf{I})$ is Gaussian noise with standard deviation σ.

Prox-JADMM solves the $\mathbf{x}_i$-subproblems with $P_i = \tau_i\mathbf{I} - \beta A_i^\top A_i$ $(i = 1, 2, \ldots, N)$ as follows:

$$\begin{aligned}
\mathbf{x}_i^{k+1} &= \arg\min_{\mathbf{x}_i} \ \|\mathbf{x}_i\|_1 + \frac{\beta}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2 + \frac{1}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|_{P_i}^2 \\
&= \arg\min_{\mathbf{x}_i} \ \|\mathbf{x}_i\|_1 + \left\langle \beta A_i^\top\left(A\mathbf{x}^k - c - \frac{\lambda^k}{\beta}\right), \mathbf{x}_i\right\rangle + \frac{\tau_i}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2.
\end{aligned} \tag{5.72}$$

Here, we choose the prox-linear P_i's to linearize the original subproblems, and thus (5.72) admits a simple closed-form solution by the *shrinkage* (or *soft-thresholding*) formula. The proximal parameters are initialized as $\tau_i = 0.1N\beta$ and are adaptively updated by the strategy discussed in Section 5.3.3.

Recall that VSADMM needs to solve the following $\mathbf{x}_i$-subproblems:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \ \|\mathbf{x}_i\|_1 + \frac{\beta}{2}\left\|A_i\mathbf{x}_i - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\beta}\right\|^2. \tag{5.73}$$

Such subproblems are not easily computable, unless $\mathbf{x}_i$ is a scalar (i.e., $n_i = 1$) or $A_i^\top A_i$ is a diagonal matrix. Instead, we solve the subproblems approximately using the prox-linear approach:

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \ \|\mathbf{x}_i\|_1 + \left\langle \beta A_i^\top \left(A_i\mathbf{x}_i^k - \mathbf{z}_i^{k+1} - \frac{c}{N} - \frac{\lambda_i^k}{\beta}\right), \mathbf{x}_i - \mathbf{x}_i^k \right\rangle + \frac{\tau_i}{2}\left\|\mathbf{x}_i - \mathbf{x}_i^k\right\|^2, \tag{5.74}$$

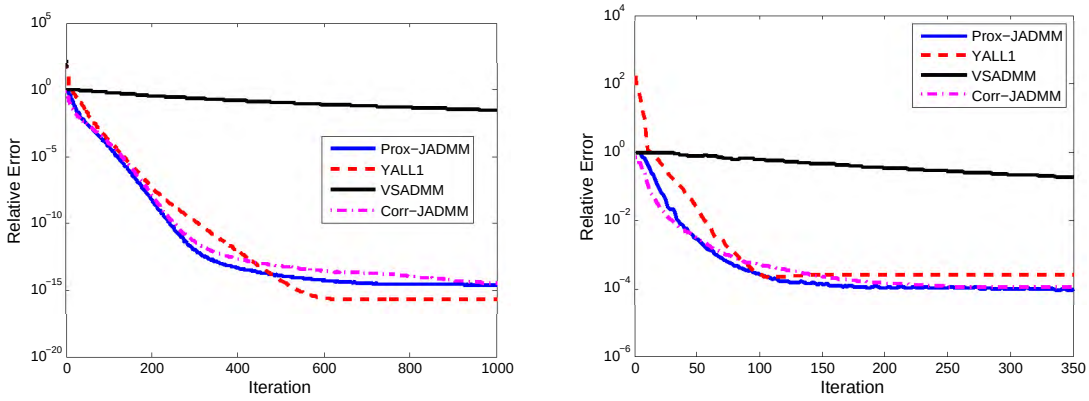
which can be easily computed by the shrinkage operator. We set $\tau_i = 1.01\beta\|A_i\|^2$ in order to guarantee the convergence, as suggested in [58].

Corr-JADMM solves the following $\mathbf{x}_i$-subproblems in the "prediction" step:

$$\tilde{\mathbf{x}}_i^{k+1} = \arg\min_{\mathbf{x}_i} \ \|\mathbf{x}_i\|_1 + \frac{\beta}{2}\left\|A_i\mathbf{x}_i + \sum_{j\neq i} A_j\mathbf{x}_j^k - c - \frac{\lambda^k}{\beta}\right\|^2. \tag{5.75}$$

Because the correction step in [33] is based on exact minimization of the subproblems, we do not apply the prox-linear approach to solve the subproblems approximately. Instead, we always partition $\mathbf{x}$ into scalar components (i.e., $N = n$) so that the subproblems (5.75) can still be computed exactly. The same penalty parameter $\beta = 10/\|c\|_1$ is used for the three algorithms. It is nearly optimal for each algorithm, selected out of a number of different values.

In addition, we also include the YALL1 package [61] in the experiment, which is one of the state-of-the-art solvers for ℓ_1 minimization. Though YALL1 is not implemented in parallel, the major computation of its iteration is matrix-vector multiplication by A and $A^\top$, which can be easily parallelized (see [48]). Since all the compared algorithms have roughly the same amount of per-iteration cost (in terms



(a) Noise-free ($\sigma = 0$) (b) Noise added ($\sigma = 10^{-3}$)

Figure 5.2 : Basis pursuit ($n = 1000,\ m = 300,\ k = 60$).

of both computation and communication), we simply let all the algorithms run for a fixed number of iterations and plot their relative error $\frac{\|\mathbf{x}^k - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2}$.

Figure 5.2 shows the comparison result where $n = 1000,\ m = 300,\ k = 60$ and the standard deviation of noise σ is set to be 0 and 10^{-3}, respectively. For Prox-JADMM and VSADMM, we set $N = 100$; for Corr-JADMM, we set $N = 1000$. The results are average of 100 random trials. We can see that Prox-JADMM and Corr-JADMM achieve very close performance and are the fastest ones among the compared algorithms. YALL1 also shows competitive performance. However, VSADMM is far slower than the others, probably due to inexact minimization of the subproblems and the conservative proximal parameters.

5.4.3 Distributed Large-Scale Basis Pursuit

In previous subsections, we described the numerical simulation of a distributed implementation of Jacobi-Proximal ADMM that was carried out in Matlab. We now turn to realistic distributed examples and solve two very large instances of the ℓ_1-

minimization problem (5.71) using a C code with MPI for inter-process communication and the GNU Scientific Library (GSL) for BLAS operations. The experiments are carried out on Amazon's Elastic Compute Cloud (EC2).

We generate two test instances as shown in Table 5.1. Specifically, a sparse solution $\mathbf{x}^*$ is randomly generated with k nonzeros drawn from the standard Gaussian distribution. Matrix A is also randomly generated from the standard Gaussian distribution with m rows and n columns, and it is partitioned evenly into $N = 80$ blocks. Vector c is then computed by $c = A\mathbf{x}^*$. Note that A is dense and has double precision. For Test 1 it requires over 150 GB of RAM and has 20 billion nonzero entries, and for Test 2 it requires over 337GB of RAM. Those two tests are far too large to process on a single PC or workstation. We want to point out that we cannot find a dataset of similar or larger size in the public domain. We are willing to test our a larger problem per reader's request.

Table 5.1 : Two large datasets

	m	n	k	RAM
dataset 1	1.0×10^5	2.0×10^5	2.0×10^3	150GB
dataset 2	1.5×10^5	3.0×10^5	3.0×10^3	337GB

We solve the problem using a cluster of 10 machines, where each machine is a "memory-optimized instance" with 68 GB RAM and 1 eight-core Intel Xeon E5-2665 CPU. Those instances run Ubuntu 12.04 and are connected with 10 Gigabit ethernet network. Since each has 8 cores, we run the code with 80 processes so that each process runs on its own core. Such a setup is charged for under $17 per hour.

We solve the large-scale ℓ_1 minimization problems with a C implementation that

matches the Matlab implementation in the previous section. The implementation consists of a single file of C code of about 300 lines, which is available for download on our personal website.

Table 5.2 : Time results for large scale basis pursuit examples

	150GB Test			337GB Test		
	Itr	Time(s)	Cost($)	Itr	Time(s)	Cost($)
Data generation	–	44.4	0.21	–	99.5	0.5
CPU per iteration	–	1.32	–	–	2.85	–
Comm. per iteration	–	0.07	–	–	0.15	–
Reach 10^{-1}	23	30.4	0.14	27	79.08	0.37
Reach 10^{-2}	30	39.4	0.18	39	113.68	0.53
Reach 10^{-3}	86	112.7	0.53	84	244.49	1.15
Reach 10^{-4}	234	307.9	1.45	89	259.24	1.22

The breakdown of the wall-clock time is summarized in Table 5.2. We can observe that Jacobi ADMM is very efficient in obtaining a relative low accuracy, which is usually sufficient for large-scale problems. We want to point out that the basic BLAS operations in our implantation can be further improved by using other libraries such as hardware-optimized BLAS libraries produced by ATLAS, Armadillo, etc. Those libraries might lead to several times of speedup*. We use GSL due to its ease of use, so the code can be easily adapted for solving similar problems.

*http://nghiaho.com/?p=1726

Chapter 6

Conclusion

In this thesis, we have mainly developed two different types of generalizations to the classic alternating direction method of multipliers (ADMM):

- **A Generalized ADMM with simplified subproblems**. The Generalized ADMM framework allows flexible ways of solving the subproblems efficiently. It subsumes many variants of ADMM such as *prox-linear ADMM*, *gradient-descent ADMM* and *ADMM with Hessian approximation*. We establish the global convergence of the Generalized ADMM and improve an existing convergence rate of $O(1/k)$ slightly to $o(1/k)$. Furthermore, we establish its linear convergence rate under various scenarios, in which at least one of the two objective functions is strictly convex and has Lipschitz continuous gradient.

- **A parallel and multi-block extension to ADMM**. We extend ADMM from the classic Gauss-Seidel setting to the Jacobi setting, from 2 blocks to N blocks. The algorithm can be implemented in a fully parallel and distributed fashion, and thus is particularly attractive for solving certain problems with huge and distributed datasets. However, the straightforward extension — *Jacobi ADMM* is not necessarily convergent. To guarantee its convergence, we provide a sufficient condition on the constraint coefficient matrices A_i. For general matrices A_i, we propose the *Jacobi-Proximal ADMM* by adding proximal terms of different kinds to the subproblems. We show that the algorithm converges globally

at a rate of $o(1/k)$, while allowing flexible and efficient ways of solving the subproblems. Numerical results are presented to demonstrate the efficiency of the proposed algorithm in comparison with several existing parallel algorithms. We also implemented our algorithm on Amazon EC2, an on-demand public computing cloud, and report its performance on very large-scale basis pursuit problems with distributed data.

These generalizations are of both theoretical and practical importance to the applications of ADMM. From the theoretical point of view, the generalizations provide unified frameworks to analyze a wide variety of ADMM-based algorithms and their variants in practice, particularly in the areas of compressive sensing, signal and image processing, machine learning and applied statistics. Our convergence analysis makes various meaningful extensions to the existing convergence theory, as well as obtaining better convergence rates.

From the practical point of view, the generalizations introduce better flexibility and efficiency in solving the subproblems, thereby making the algorithms more scalable for many large-scale problems. Our analysis provides a deeper understanding of the underlying convergence mechanism of ADMM. It sheds lights on how to design and tune the algorithms in practice. In particular, our derived linear convergence rates provide insights on how the penalty parameter β affects the convergence speed, thereby providing some theoretical guidance for choosing β. Also, the convergence analysis of the Jacobi-Proximal ADMM enables us to design a way of adaptively tuning the proximal parameters to make its convergence much faster.

Besides the generalizations discussed in this thesis, we leave many other generalizations and analyses for future study. We list a few topics below:

- In both of the Generalized ADMM and the Jacobi-Proximal ADMM frame-

works, the proximal terms in the subproblems are restricted to be *quadratic* terms. The quadratic proximal term is perhaps most commonly used and is often simple enough to compute. However, we may make further generalization by allowing *non-quadratic* proximal terms (such as Bregman divergence). In some applications, choosing certain non-quadratic proximal terms wisely may make the subproblems simpler to solve and make the algorithm run faster than using quadratic proximal terms. Further research is needed to understand the convergence behaviors when using non-quadratic proximal terms and how to choose these proximal terms in practice.

- The penalty parameter β and the proximal parameters P/Q (in Generalized ADMM) or $P_i,\ i = 1, 2, \ldots, N$ (in Jacobi-Proximal ADMM) are important parameters affecting the performance of the algorithms. To make the presentation of our analysis simpler to understand, we have assumed these parameters to be fixed over iterations for most part of the thesis. Empirical studies have shown that updating these parameters properly during the iterations can often greatly improve the performance of the algorithm and make it more robust to the initial values of the parameters. In Section 5.3.3, we have briefly discussed a simple method for adjusting the proximal parameters of the Jacobi-Proximal ADMM, while the penalty parameter is assumed to be fixed. It remains open how to effectively update the penalty parameter in coordination with the adjustment of proximal parameters. It still needs further investigation and improvement.

- For the Jacobi-Proximal ADMM, we have established its $o(1/k)$ rate of convergence under fairy mild conditions. Analogous to our linear convergence results for the Generalized ADMM, we should also be able to show the linear conver-

gence of the Jacobi-Proximal ADMM under certain additional assumptions such as strong convexity and Lipschitz continuous gradients of the objective functions. Also, similar results might be obtained for the Gauss-Seidel multi-block extension of ADMM.

- Though the classic ADMM can be considered as a special case of the Douglas-Rachford Splitting Method (DRSM), it is unclear whether one can establish equivalence between our extensions of ADMM (i.e., the Generalized ADMM and the Jacobi-Proximal ADMM) and any operator splitting methods with skillfully chosen operators

- In addition, there is a variety of interesting ongoing development of ADMM such as the *stochastic ADMM* [45], *online ADMM* [54, 57], and *asynchronous ADMM* [60]. ADMM seems to be a powerful and promising algorithmic tool for many modern "Big Data" applications which will remain for future research and development.

Bibliography

[1] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods, Second Edition*, Athena Scientific, 1997.

[2] D. P. Bertsekas, *Extended monotropic programming and duality*, Journal of optimization theory and applications, 139 (2008), pp. 209–225.

[3] D. Boley, *Linear convergence of ADMM on a model problem*, TR 12-009, Department of Computer Science and Engineering, University of Minnesota, (2012).

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Machine Learning, 3 (2010), pp. 1–122.

[5] Y. Boykov and V. Kolmogorov, *An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26 (2004), pp. 1124–1137.

[6] J. F. Cai, S. Osher, and Z. Shen, *Split Bregman methods and frame based image restoration*, Multiscale modeling & simulation, 8 (2009), p. 337.

[7] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Foundations of Computational mathematics, 9 (2009), pp. 717–772.

[8] V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky, *Latent variable graphical model selection via convex optimization*, Annals of Statistics, 40 (2012),

pp. 1935–1967.

[9] C. H. Chen, B. S. He, Y. Y. Ye, and X. M. Yuan, *The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent*, preprint, (2013).

[10] G. Chen and M. Teboulle, *A proximal-based decomposition method for convex minimization problems*, Mathematical Programming, 64 (1994), pp. 81–101.

[11] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM review, 43 (2001), pp. 129–159.

[12] E. Corman and X. M. Yuan, *A generalized proximal point algorithm and its convergence rate*, (2013).

[13] W. Deng and W. Yin, *On the global and linear convergence of the generalized alternating direction method of multipliers*, Rice University CAAM Technical Report TR12-14, (2012).

[14] W. Deng, W. Yin, and Y. Zhang, *Group sparse optimization by alternating direction method*, TR11-06, Department of Computational and Applied Mathematics, Rice University, (2011).

[15] J. Douglas and H. Rachford, *On the numerical solution of heat conduction problems in two and three space variables*, Transactions of the American mathematical Society, 82 (1956), pp. 421–439.

[16] J. Eckstein and D. P. Bertsekas, *An alternating direction method for linear programming*, (1990).

[17] J. Eckstein and D. P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Mathematical Programming, 55 (1992), pp. 293–318.

[18] E. Esser, *Applications of Lagrangian-based alternating direction methods and connections to split Bregman*, CAM report 09-31, UCLA, (2009).

[19] H. Everett, *Generalized lagrange multiplier method for solving problems of optimum allocation of resources*, Operations research, 11 (1963), pp. 399–417.

[20] D. Gabay, *Chapter ix applications of the method of multipliers to variational inequalities*, Studies in mathematics and its applications, 15 (1983), pp. 299–331.

[21] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers & Mathematics with Applications, 2 (1976), pp. 17–40.

[22] R. Glowinski, *Numerical methods for nonlinear variational problems*, Springer Series in Computational Physics, 1984.

[23] R. Glowinski and A. Marrocco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires*, Laboria, 1975.

[24] D. Goldfarb and S. Ma, *Fast multiple splitting algorithms for convex optimization*, SIAM Journal on Optimization, 22 (2012), pp. 533–556.

[25] D. Goldfarb, S. Ma, and K. Scheinberg, *Fast alternating linearization methods for minimizing the sum of two convex functions*, Mathematical Programming, (2010), pp. 1–34.

[26] D. Goldfarb and W. Yin, *Parametric maximum flow algorithms for fast total variation minimization*, SIAM Journal on Scientific Computing, 31 (2009), pp. 3712–3743.

[27] T. Goldstein, X. Bresson, and S. Osher, *Geometric applications of the split Bregman method: Segmentation and surface reconstruction*, Journal of Scientific Computing, 45 (2010), pp. 272–293.

[28] T. Goldstein, B. O'Donoghue, and S. Setzer, *Fast alternating direction optimization methods*, CAM report 12-35, UCLA, (2012).

[29] T. Goldstein and S. Osher, *The split Bregman method for ℓ_1 regularized problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 323–343.

[30] B. He, L. Z. Liao, D. Han, and H. Yang, *A new inexact alternating directions method for monotone variational inequalities*, Mathematical Programming, 92 (2002), pp. 103–118.

[31] B. S. He, *A class of projection and contraction methods for monotone variational inequalities*, Applied Mathematics and Optimization, 35 (1997), pp. 69–76.

[32] ——, *Parallel splitting augmented lagrangian methods for monotone structured variational inequalities*, Computational Optimization and Applications, 42 (2009), pp. 195–212.

[33] B. S. He, L. S. Hou, and X. M. Yuan, *On full Jacobian decomposition of the augmented lagrangian method for separable convex programming*, (2013).

[34] B. S. He, M. Tao, and X. M. Yuan, *Alternating direction method with gaussian back substitution for separable convex programming*, SIAM Journal on

Optimization, 22 (2012), pp. 313–340.

[35] B. S. He and X. M. Yuan, *On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers*, (2012).

[36] ——, *On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method*, SIAM Journal on Numerical Analysis, 50 (2012), pp. 700–709.

[37] M. R. Hestenes, *Multiplier and gradient methods*, Journal of optimization theory and applications, 4 (1969), pp. 303–320.

[38] M. Hong and Z.-Q. Luo, *On the linear convergence of the alternating direction method of multipliers*, arXiv preprint arXiv:1208.3922, (2012).

[39] H. Jiang, W. Deng, and Z. Shen, *Surveillance video processing using compressive sensing*, Inverse Problems and Imaging, 6 (2012).

[40] P.-L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 964–979.

[41] G. Mateos, J. A. Bazerque, and G. B. Giannakis, *Distributed sparse linear regression*, IEEE Transactions on Signal Processing, 58 (2010), pp. 5262–5276.

[42] J. M. Mendel and C. S. Burrus, *Maximum-likelihood deconvolution: a journey into model-based signal processing*, Springer-Verlag New York, 1990.

[43] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, *D-ADMM: A communication-efficient distributed algorithm for separable optimization*, IEEE Transactions on Signal Processing, 61 (2013), pp. 2718–2723.

[44] Y. Nesterov and I. E. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer, 2004.

[45] H. Ouyang, N. He, L. Tran, and A. Gray, *Stochastic alternating direction method of multipliers*, in Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 80–88.

[46] N. Parikh and S. Boyd, *Block splitting for distributed optimization*, Mathematical Programming Computation, (2013), pp. 1–26.

[47] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, *RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2012), pp. 2233–2246.

[48] Z. Peng, M. Yan, and W. Yin, *Parallel and distributed sparse optimization*, IEEE Asilomar Conference on Signals Systems and Computers, (2013).

[49] M. J. Powell, *A method for non-linear constraints in minimization problems*, UKAEA, 1967.

[50] R. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM Journal on Control and Optimization, 14 (1976), pp. 877–898.

[51] R. T. Rockafellar, *Monotropic programming: descent algorithms and duality*, Nonlinear programming, 4 (1981), pp. 327–366.

[52] ——, *Convex analysis*, vol. 28, Princeton University Press, 1997.

[53] N. Z. Shor, K. C. Kiwiel, and A. Ruszcayski, *Minimization methods for non-differentiable functions*, Springer-Verlag New York, Inc., 1985.

[54] T. Suzuki, *Dual averaging and proximal gradient descent for online alternating direction multiplier method*, in Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 392–400.

[55] M. Tao and X. M. Yuan, *Recovering low-rank and sparse components of matrices from incomplete and noisy observations*, SIAM Journal on Optimization, 21 (2011), pp. 57–81.

[56] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society. Series B (Methodological), (1996), pp. 267–288.

[57] H. Wang and A. Banerjee, *Online alternating direction method*, in Proceedings of the 29th International Conference on Machine Learning (ICML-12), 2012, pp. 1119–1126.

[58] X. F. Wang, M. Y. Hong, S. Q. 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).

[59] Y. Wang, J. Yang, W. Yin, and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 248–272.

[60] E. Wei and A. Ozdaglar, *On the $O(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers*, arXiv preprint arXiv:1307.8254, (2013).

[61] J. F. Yang and Y. Zhang, *Alternating direction algorithms for ℓ_1-problems in compressive sensing*, SIAM journal on scientific computing, 33 (2011), pp. 250–278.

[62] X. Zhang, M. Burger, and S. Osher, *A unified primal-dual algorithm framework based on Bregman iteration*, Journal of Scientific Computing, 46 (2011), pp. 20–46.

[63] H. Zou and T. Hastie, *Regularization and variable selection via the elastic net*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67 (2005), pp. 301–320.