Large Scale Optimization for Machine Learning

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

Huahua Wang

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Arindam Banerjee

December, 2014

Acknowledgements

There are many people that have earned my gratitude for their contribution to my time in graduate school.

First and foremost, I would like to express my sincerest gratitude and appreciation to my advisor Prof. Arindam Banerjee for his invaluable support and guidance throughout my graduate study. His motivation, enthusiasm, immense knowledge and dedication to research trigger my interests on research. His advice has been extraordinary by all means, opening the door of machine learning for me, letting me fly, introducing me to professors and senior researchers, encouraging me to reach others, sharpening my technical presentation and writing skills, and overall preparing me to be an independent and competent researcher.

Second, I would like to extend my great thanks to Prof. Daniel Boley, Vipin Kumar, Xiaotong Shen to serve on my thesis committee and Prof. Zhiquan Luo and Yousef Saad on the committee of my thesis proposal.

Third, I am also grateful to my collaborators: Daniel Boley, Inderjit Dhillon, Qiang Fu, Chen Jin, Shiva Prasad Kasiviswanathan, Cho-Jui Hsieh, Zhiquan Luo, Prem Melville, Pradeep Ravikumar. It was my honor to work with them. I would also thank my collegues in the machine learning department in the NEC Labs, Princeton, where I ventured into deep learning, and particularly Dr. Martin Renqiang Min who offered me the opportunity.

Last but not least, I would also like to show my gratitude to my lab mates: Amrudin Agovic, Soumyadeep Chatterjee, Sheng Chen, Konstantina Christakopoulou, Puja Das, Farideh Fazayeli, Qiang Fu, Andre Goncalves, Nicholas Johnson, Igor Melnyk, Hanhuai Shan, Vidyashankar Sivakumar, Karthik Subbian, Amir Taheri and Tinghui Zhou. Thank you for teaching me diversified culture, languages, etc.

Dedication

To my parents.

Abstract

Over the last several decades, tremendous tools have been developed in machine learning, ranging from statistical models to scalable algorithms, from learning strategies to various tasks, having a far-reaching influence in broad applications ranging from image and speech recognitions to recommender systems, and from bioinformatics to robotics. In entering the era of big data, large scale machine learning tools become increasingly important in training a big model on a big dataset. Since machine learning problems are fundamentally empirical risk minimization problems, large scale optimization plays a key role in building a large scale machine learning system. However, scaling classical optimization algorithms like stochastic gradient descent (SGD) in a distributed system raises some issues, e.g., synchronization. Synchronization is required because consistency should be maintained, i.e., the parameters in different machines should be the same. Synchronization leads to blocking computation and performance degradation of a distributed system. Without blocking, overwriting may happen and consistency can not be guaranteed. Moreover, SGD may not be suitable for constrained optimization problems.

To address the issues of scaling optimization algorithms, we develop several novel optimization algorithms suitable for distributed systems from two settings, i.e., unconstrained optimization and equality-constrained optimization. First, building on SGD in the unconstrained optimization setting, we propose online randomized block coordinate descent which randomly updates some parameters using some samples and thus allows the overwriting in SGD. Second, instead of striving to maintain consistency at each iteration in the unconstrained optimization setting, we turn to the equality-constrained optimization which guarantees eventual consistency , i.e., the parameters in different machines are not the same at each iteration but will be the same eventually. The equality-constrained optimization also includes the cases that SGD can not be applied.

The alternating direction method of multipliers (ADMM) provides a suitable framework for equality-constrained optimziation but raises some issues: (1) it does not provide a systematic way to solve subproblems; (2) it requires to solve all subproblems and synchronization; (3) it is a batch method which can not process data online. For the first issue, we propose Bregman ADMM which provides a unified framework to solve subproblems efficiently. For the second issue, we propose parallel direction method of multipliers (PDMM), which randomly picks some subproblems to solve and does asynchronous aggregation. Finally, we introduce online ADMM so that the algorithm can process partial data at each iteration.

To validate the effectiveness and scalability of the proposed algorithms, we particularly apply them to a variety of applications, including sparse structure learning and maximum a posterior (MAP) inference in probabilistic graphical models, and online dictionary learning. We also implement the proposed methods on various architectures, including hundreds to thousands CPU cores in clusters and GPUs. Experimental results show that the proposed methods can scale gracefully with the number of cores and perform better than state-of-the-art methods.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, the amount of data being collected in a variety of scientific, societal, and commercial domains has exploded at a rate we have never seen before. These include search engines, online social networks, online gaming systems, healthcare, social media, climate sciences, ecology and environmental sciences, finance and economics and so on. The trend is expected to continue, in fact grows several folds, in the foreseeable future. In the past decade, we have seen great success in efficiently managing the massive data with the rise of Google, Facebook, Baidu, Alibaba and many other internet companies. In entering a new stage of gaining intelligence or knowledge from learning a large amount of data, *machine learning* has earned its reputation after many years of competition [91, 96] on the way towards the so-called *artificial intelligence*. In fact, we are seeing the rise of machine learning in industry, e.g., Siri in Apple iphone, Google brain, face recognition in Facebook, letting alone the use of pervasive recommender systems in Netflix and Amazon. In the foreseeable future, machine learning will reshape the business in more and more sectors like healthcare, retail and manufacturing, as happening in the Internet sector.

In principle, machine learning consists of three components, i.e., data, model and algorithm. For a particular task, machine learning is to learn a model from data using some algorithm. Although the data and model could be very different for different applications, the underlying problems required to be solved in machine learning are strikingly similar. As shown by statistical learning theory [195, 110, 135, 18, 9, 138, 102, 103](see Section 1.1), most machine learning problems can be reduced to solving *empirical risk minimization* problems or *structural risk minimization* problems, which often resort to *optimization*.

In the era of big data, model also needs to grow bigger, in order to accomodate the complexity of big data. For example, an image recognition system named Google brain [42], which can recognize objects like birds, cats and human in images, needs to train a large neural network model with 4 billion parameters on 16 million images. Deepface [184] trains a deep network with more than 120 million parameters on 4 million facial images. However, the big model and data poses a great challenge to the learning, which may take weeks or months if running on a single machine and using conventional algorithms. To accelerate the learning, parallelism is indespensable. Two kinds of parallelism are commonly considered. One is *model parallelism* where model is distributed across multiple devices and trained in parallel. The other is *data parallelism* where data is distributed across multiple devices and processed in parallel. The partition of data and model raises issues of synchronization when deploying current optimization algorithms which are usually designed for a single machine to a distributed system. Synchronization is to guarantee that model parameters in different machines are the same, which are called *consistency*. Synchronization leads to blocking computation, which greatly degrades the performance of a distributed system. Therefore, we need to design novel optimization algorithms suitable for a distributed system. How to scale current optimization algorithms in a distributed system is the main motivation of this thesis, which will be discussed in detail in Section 1.2.

1.1 Statistical Learning Theory

Given an input space $\mathcal{X}$, an output space $\mathcal{Y}$ and a probability distribution $P(\mathbf{x}, y)$ of input-output pairs $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$. Consider a family of classifiers $\mathcal{F}$, consisting of real-valued functions defined on the space Ω, and assume that each $f \in \mathcal{F}$ maps Ω to $\mathcal{Y}$. To measure the goodness of the classifier, a loss function $\ell(f(\mathbf{x}), y) : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is defined. In classification, $\ell(f(\mathbf{x}), y) = \mathbb{1}(f(\mathbf{x}) \neq y)$ where $\mathbb{1}$ is the indicator function. The expectation of the loss over distribution $P(\mathbf{x}, \mathbf{y})$ is defined as *expected risk*, i.e.,

$$E(f) = \mathbb{E}_{P(\mathbf{x},y)}[\ell(f(\mathbf{x}), y)] = \int \ell(f(\mathbf{x}), y) dP(\mathbf{x}, y) \ . \tag{1.1}$$

Since $P(\mathbf{x}, \mathbf{y})$ is unknown, the expected risk is unknown. In practice, we are given a set of training examples $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$, which are assumed as independent and identically distributed (i.i.d.) random samples of $P(\mathbf{x}, \mathbf{y})$. The average loss over n training samples is defined as *empirical risk*,

$$E_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\mathbf{x}_i), y_i) \; . \tag{1.2}$$

The *generalization error* of $E_n(f)$ is used to characterize the goodness of a model learned from training samples, , which is defined as

$$\max_{f \in \mathcal{F}} \; E(f) - E_n(f) \; . \tag{1.3}$$

For any model $f \in \mathcal{F}$, if (1.3) is well bounded by some metrics, the performance of the learned model on the unseen samples is guaranteed. The last several decades have witnessed the establishment of mathematical foundations of statistical learning theory on the *generalization error bound* of a predicative function $f \in \mathcal{F}$ in (1.3) (e.g. classifier) learned from data applied to the unknown observations [195, 110, 135, 18, 9, 138, 102, 103]. The *generalization error bound* is usually characterized by *Vapnik-Chervonenkis dimension* or *VC-dimension* [195] which measures the complexity of a function class $\mathcal{F}$. In binary classification, for any f, with high probability, we have

$$E(f) \leq E_n(f) + c\sqrt{\frac{\text{VCdim}(\mathcal{F})}{n}} \; , \tag{1.4}$$

where c is an constant and VCdim$(\mathcal{F})$ denotes the VC-dimension of class $\mathcal{F}$ [9].

As a result, many machine learning problems are fundamentally solving the following *empirical risk minimization* problem:

$$\min_{f \in \mathcal{F}} E_n(f) \; . \tag{1.5}$$

To avoid the *overfitting*, a regularizer $R(f)$ is often used to control the complexity of model, then we have the following *structural risk minimization* problem:

$$\min_{f \in \mathcal{F}} E_n(f) + \lambda R(f) \; , \tag{1.6}$$

where $\lambda > 0$ is the regularization parameter. Most of machine learning methods are in the form of (1.6), including support vector machines (SVM) [38, 39], sparse logistic regression [61] and deep learning [81, 46]. This thesis will revolve around how to efficiently solve the empirical risk minimization problem using some optimization methods.

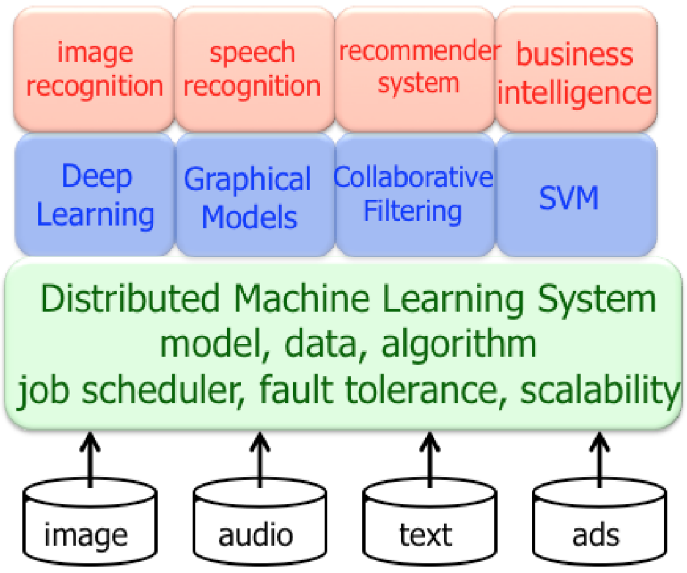


Figure 1.1: Four layers of distributed machine learning system: (a) data layer, (b) system layer, (c) model layer, (d) application layer.

1.2 Distributed Machine Learning System

As both data and model grow bigger and bigger, the empirical risk minimization problem (1.5) could consist of millions to billions of data points and that amount of model parameters, posing a great challenge to solving the optimization problem. Therefore, considerable efforts have been devoted to scaling up optimization algorithms through parallalization and distribution [42, 113, 112, 40, 226, 150, 123, 64] and on various architectures, including shared-memory architectures [150], distributed memory architectures [112, 40, 42, 64] and GPUs [157].

In general, a distributed machine learning system is made up of four layers, i.e., data layer, system layer, model layer and application layer, as illustrated in Figure 1.1. In data layer, data may store in multiple machines and will be fed into the system asynchronously. System layer manages all kinds of resources like data, model and computing infrastructures. It should provide dynamic job scheduling, efficient communication, fault tolerance and elastic scalability. A key component in the system layer is optimization or learning algorithms which conduct resources to train a model. Model layer defines common machine learning models or methods, e.g., deep learning [81, 46], probabilistic graphical models [198, 101], SVM [38, 39] and so on. Once a model has been trained, we may use the model to recognize images or recommend products, which are going to be defined in the application layer.

First generation distributed system simply uses MapReduce [43], e.g. Mahout in Hadoop. Since MapReduce is not efficient for iterative-convergent optimization algorithms, Spark[1] and Graphlab [123] have then been developed. Very recently, *Parameter Server* [42, 112, 113, 40] has been proposed as a general-purpose distributed machine learning system. In the parameter server, model parameters are distributed across several servers and data can only be accessed locally by computing machines or the so-called workers. While servers maintain global parameters, each worker has a local copy of global parameters and local data. In addition, workers perform the following three operations:

- Pull the global parameters from the servers;
- Compute local results using the copy of parameters and local data;
- Push local results to the servers.

The parameter server usually uses the optimization algorithm called *stochastic gradient descent* (SGD) [162, 95] or *online gradient descent* (OGD) [27, 225, 76, 52, 53, 210] . However, SGD requires synchronization in the first step and the third step because the algorithm should use consistent parameters to make progress. All workers pull the newest and same parameters from the servers. The servers should gather all results from workers to update global parameters, otherwise the results of slower workers will be discarded. With synchronization, consistency is guaranteed but the computations should be blocked or locked, e.g., fast workers should wait slower workers. However, synchronization will slow down the system, particularly considering that jobs may fail or may be preempted by the system. To avoid the blocking, the parameter server [42] was implemented without synchronization, but runs the risk of encountering the following two issues:

- Some worker uses old parameters while other workers uses new parameters;
- Several workers compete to write results to the server, leading to overwriting.

However, it is still remain unclear that whether SGD with such changes is guaranteed to work. In addition to the issue of consistency in the parameter server, parameter server is mainly designed for unconstrained optimization problems, which may not be suitable for constrained

[1] http://spark.apache.org/

optimization problems, e.g., MAP inference in probabilistic graphical models [198] and sparse structure learning [23].

1.3 Contributions and Organization of the Thesis

The main contribution of this thesis is to develop novel optimization methods to address the issues encountered in a distributed system. Nonetheless, we prove the rate of the convergence of the proposed methods. The thesis is divided into three main parts: (1) unconstrained optimization; (2) equality-constrained optimization; (3) applications. We describle each part in detail.

Part I is to address the overwriting issue of SGD in the parameter server. When the first worker overwrites the result of the second worker, the parameters in servers are only updated by the first worker, implying that part of model parameters are randomly updated by some workers using part of data.

- In Chapter 2, we propose online randomized block coordinate descent (ORBCD) [202] by combining the two well-known algorithms named stochastic gradient descent (SGD) and randomzied block coordinate descent. At each iteration, ORBCD only computes the partial gradient of one block coordinate of one mini-batch samples. ORBCD is well suited for the composite minimization problem where one function is the average of the losses of a large number of samples and the other is a simple regularizer defined on high dimensional variables. We show that the iteration complexity of ORBCD has the same order as OGD or SGD. For strongly convex functions, by reducing the variance of stochastic gradients, we show that ORBCD can converge at a geometric rate in expectation, matching the convergence rate of SGD with variance reduction and RBCD.

Part II focuses on the issue of consistency and constrained optimization problems under the study of the equality-constrained optimization problems. The equality constraints may come from the problems themselves, e.g., MAP inference in probabilistic graphical models [198] and sparse structure learning [23], or may come from the decomposition of the big empirical loss minimization problems into many small problems in a distributed system. For the later case, each subproblem maintains a local (partial) copy of global model parameters. To guarantee the consistency between the local model and the global model, an inequality constraint is introduced, leading to the following equality-constrained optimization problems:

$$\min_{\mathbf{x}} \sum_{j=1}^{J} f_j(\mathbf{x}_j) + g(\mathbf{z}) \qquad \text{s.t.} \quad \mathbf{x}_j = \mathbb{U}_j \mathbf{z} \, , \tag{1.7}$$

where $\mathbf{x}_j$ is a local model parameter, $\mathbf{z}$ is the global model parameter, $\mathbb{U}_j$ is a diagonal matrix where diagonal elements are 1 or 0, serving to choose the coordinates of $\mathbf{z}$. We do not intend to maintain consistency at each iteration as SGD, i.e., the equality constraint is always satisfied. Instead, the inconsistency is allowed at each iteration but the consistency can be achieved eventually.

The alternating direction method of multipliers (ADMM) provides a suitable framework for the equality-constrained optimization problems but raises some issues. ADMM consists of three main steps: (1) split the big problem (global model with all datasets) into many small subproblems (local model with partial datasets), (2) solve all subproblems in parallel, (3) synchronize the results of all subproblems. In the first step, it is important to achieve a tradeoff between the number of subproblems and how efficiently the subproblems can be solved. As the number of subproblems increases, communication increases and the convergence will slow down. On the other hand, reducing the number of subproblems will increase the complexity of subproblems, e.g., whether the subproblems can be efficiently solved. To this end, we propose Bregman ADMM where the subproblems can be solved efficiently. In the second and third step, if there is a large number of subproblems, solving all subproblems and synchronizing all of them make ADMM unappealing compared to SGD. To address the two issues, we propose parallel direction method of multipliers (PDMM), which randomly picks some subproblems to solve and does asynchronous aggregation. Finally, we propose online ADMM which process partial data at each iteration. The organizations of chapters in Part II are described as follows:

- In Chapter 3, we first briefly review the alternating direction method of multipliers (ADMM) for solving the equality-constrained optimization problems. Then we prove the rate of convergence of ADMM [200], facilitating the improvement and modifications of ADMM which are needed in some scenarios.

- In Chapter 4, we generalize ADMM to Bregman ADMM (BADMM) [201], which allows the choice of different Bregman divergences to exploit the structure of problems.

BADMM provides a unified framework for ADMM and its variants, including generalized ADMM, inexact ADMM and Bethe ADMM. We establish the global convergence and the $O(1/T)$ iteration complexity for BADMM. In some cases, BADMM can be faster than ADMM by a factor of $O(n/\log(n))$ where n is the dimension of the problem. Experimental results are illustrated on the mass transportation problem, which can be solved in parallel by BADMM. BADMM is faster than ADMM and highly optimized commercial software Gurobi, particularly when implemented on GPU.

- In Chapter 5, we propose a parallel randomized block coordinate variant of ADMM named parallel direction method of multipliers (PDMM) [204, 205] to solve the optimization problems with multi-block linear constraints. PDMM can randomly update primal and dual blocks in parallel, behaving like parallel randomized block coordinate descent. We establish the global convergence and the iteration complexity for PDMM with constant step size. We also show that PDMM can do randomized block coordinate descent on overlapping blocks. Experimental results show that PDMM performs better than state-of-the-arts methods in two applications, robust principal component analysis and overlapping group lasso.
- Chapter 6, we generalize ADMM to the online and stochastic setting where ADMM can simply work on a mini-batch samples at each iteration [200]. We consider two scenarios in the online setting, based on whether an additional Bregman divergence is needed or not. In both settings, we establish regret bounds for both the objective function as well as constraints violation for general and strongly convex functions.

Part III contains several applications of proposed algorithms to a variety of machine learning problems. In particular, Bregman ADMM is used to do large scale sparse structure learning and MAP inference in probabilistic graphical models.

- Chapter 7 presents Bethe-ADMM [64] to solve the maximum a posteriori (MAP) inference in Markov Random Fields (MRF). Bethe-ADMM is based on two ideas: tree-decompositon of the graph and Bregman ADMM. The Bregman divergence used in Bethe-ADMM is induced by negative Bethe-entropy, which makes the tree subproblem easy to solve in parallel using the sum-product algorithm. The proposed algorithm is extensively evaluated on both synthetic and real datasets to illustrate its effectiveness. Further, the parallel Bethe-ADMM is shown to scale almost linearly with increasing number of cores [92].

- In Chapter 8, we consider a Bregman ADMM (inexact ADMM) algorithm for the problem of sparse precision matrix estimation in high dimensions using the (CLIME) estimator, which has several desirable theoretical properties. We develop a large scale distributed framework for the computations, which scales to millions of dimensions and trillions of parameters, using hundreds of cores [203]. The proposed framework solves CLIME in column-blocks and only involves elementwise operations and parallel matrix multiplications. We evaluate our algorithm on both shared-memory and distributed-memory architectures, which can use block cyclic distribution of data and parameters to achieve load balance and improve the efficiency in the use of memory hierarchies. Experimental results show that our algorithm is substantially more scalable than state-of-the-art methods and scales almost linearly with the number of cores.

- In Chapter 9, following Chapter 8, we generalize the CLIME estimator to double plugin Gaussian (DoPinG) copula estimators for the problem of estimating sparse precision matrix of Gaussian copula distributions using samples with missing values in high dimensions [206]. DoPinG uses two plugin procedures and consists of three steps: (1) estimate nonparametric correlations based on observed values, including Kendall's tau and Spearman's rho; (2) estimate the non-paranormal correlation matrix; (3) plug into existing sparse precision estimators. We prove that DoPinG copula estimators consistently estimate the non-paranormal correlation matrix at a rate of $O(\frac{1}{(1-\delta)}\sqrt{\frac{\log p}{n}})$, where δ is the probability of missing values. We provide experimental results to illustrate the effect of sample size and percentage of missing data on the model performance. Experimental results show that DoPinG is significantly better than estimators like mGlasso, which are primarily designed for Gaussian data.

- In Chapter 10, online ADMM is used to solve the problem of online ℓ_1-dictionary learning [98]. Online ℓ_1-dictionary learning uses the ℓ_1-penalty to measure the reconstruction error, in contrast to the squared loss in traditional dictionary learning. Online ℓ_1-dictionary learning is particularly effective in the automated identification of such news is the detection of novel documents from a voluminous stream of text documents in a robust and scalable manner. Empirical results on news-stream and Twitter data, shows that this online ℓ_1-dictionary learning algorithm for novel document detection gives more than an order of magnitude speedup over the previously known batch algorithm, without any significant loss in quality of results.

Finally, we conclude the thesis in Chapter 11 by summarizing the contributions of thesis and discussing future work.

Part I

Unconstrained Optimization

Chapter 2

Online Randomized Block Coordinate Descent

2.1 Introduction

In recent years, considerable efforts in machine learning have been devoted to solving the following composite objective minimization problem:

$$\min_{\mathbf{x}} \ f(\mathbf{x}) + g(\mathbf{x}) = \frac{1}{I}\sum_{i=1}^{I} f_i(\mathbf{x}) + \sum_{j=1}^{J} g_j(\mathbf{x}_j) \ , \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^{n\times 1}$ and $\mathbf{x}_j$ is a block coordinate of $\mathbf{x}$. $f(\mathbf{x})$ is the average of some smooth functions, and $g(\mathbf{x})$ is a *simple* function which may be non-smooth. In particular, $g(\mathbf{x})$ is block separable and blocks are non-overlapping. A variety of machine learning and statistics problems can be cast into the problem (2.1). In regularized risk minimization problems [74], f is the average of losses of a large number of samples and g is a simple regularizer on high dimensional features to induce structural sparsity [4]. While f is separable among samples, g is separable among features. For example, in lasso [189], f_i is a square loss or logistic loss function and $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ where λ is the tuning parameter. In group lasso [218], $g_j(\mathbf{x}_j) = \lambda\|\mathbf{x}_j\|_2$, which enforces group sparsity among variables. To induce both group sparsity and sparsity, sparse group lasso [61] uses composite regularizers $g_j(\mathbf{x}_j) = \lambda_1\|\mathbf{x}_j\|_2 + \lambda_2\|\mathbf{x}_j\|_1$ where λ_1 and λ_2 are the tuning parameters.

Due to the simplicity, gradient descent (GD) type methods have been widely used to solve problem (2.1). If g_j is nonsmooth but simple enough for *proximal mapping*, it is better to just use the gradient of f_i but keep g_j untouched in GD. This variant of GD is often called proximal splitting [37] or proximal gradient descent (PGD) [191, 11] or forward/backward splitting method (FOBOS) [53]. Without loss of generality, we simply use GD to represent GD and its variants in the rest of this chapter. Let m be the number of samples and n be dimension of features. m samples are divided into I blocks (mini-batch), and n features are divided into J non-overlapping blocks. If both m and n are large, solving (2.1) using batch methods like gradient descent (GD) type methods is computationally expensive. To address the computational bottleneck, two types of low cost-per-iteration methods, online/stochastic gradient descent (OGD/SGD) [162, 95, 27, 225, 76, 52, 53, 210] and randomized block coordinate descent (RBCD) [148, 21, 161, 160], have been rigorously studied in both theory and applications.

Instead of computing gradients of all samples in GD at each iteration, OGD/SGD only computes the gradient of one block samples, and thus the cost-per-iteration is just one I-th of GD. For large scale problems, it has been shown that OGD/SGD is faster than GD [185, 175, 176]. OGD and SGD have been generalized to handle composite objective functions [146, 37, 191, 11, 52, 53, 210]. OGD and SGD use a decreasing step size and converge at a slower rate than GD. In stochastic optimization, the slow convergence speed is caused by the variance of stochastic gradients due to random samples, and considerable efforts have thus been devoted to reducing the variance to accelerate SGD [168, 174, 211, 93, 126, 221]. Stochastic average gradient (SVG) [168] is the first SGD algorithm achieving the linear convergence rate for stronly convex functions, catching up with the convergence speed of GD [145]. However, SVG needs to store all gradients, which becomes an issue for large scale datasets. It is also difficult to understand the intuition behind the proof of SVG. To address the issue of storage and better explain the faster convergence, [93] proposed an explicit variance reduction scheme into SGD. The two scheme SGD is refered as stochastic variance reduction gradient (SVRG). SVRG computes the full gradient periodically and progressively mitigates the variance of stochastic gradient by removing the difference between the full gradient and stochastic gradient. For smooth and strongly convex functions, SVRG converges at a geometric rate in expectation. Compared to SVG, SVRG is free from the storage of full gradients and has a much simpler proof. The similar idea was also proposed independently by [126]. The results of SVRG is then improved in [105]. In [211], SVRG is generalized to solve composite minimization problem by incorporating the variance reduction technique into proximal gradient method.

On the other hand, RBCD [148, 160, 124, 176, 30, 90, 115] has become increasingly popular due to high dimensional problem with structural regularizers. RBCD randomly chooses a block coordinate to update at each iteration. The iteration complexity of RBCD was established in [148], improved and generalized to composite minimization problem by [160, 124]. RBCD can choose a constant step size and converge at the same rate as GD, although the constant is usually J times worse [148, 160, 124]. Compared to GD, the cost-per-iteration of RBCD is much cheaper. Block coordinate descent (BCD) methods have also been studied under a deterministic cyclic order [171, 190, 125]. Although the convergence of cyclic BCD has been established [190, 125], the iteration of complexity is still unknown except for special cases [171].

While OGD/SGD is well suitable for problems with a large number of samples, RBCD is suitable for high dimension problems with non-overlapping composite regularizers. For large scale high dimensional problems with non-overlapping composite regularizers, it is not economic enough to use one of them. Either method alone may not suitable for problems when data is distributed across space and time or partially available at the moment [148]. In addition, SVRG is not suitable for problems when the computation of full gradient at one time is expensive. In this chapter, we propose a new method named online randomized block coordinate descent (ORBCD) which combines the well-known OGD/SGD and RBCD together. ORBCD first randomly picks up one block samples and one block coordinates, then performs the block coordinate gradient descent on the randomly chosen samples at each iteration. Essentially, ORBCD performs RBCD in the online and stochastic setting. If f_i is a linear function, the cost-per-iteration of ORBCD is $O(1)$ and thus is far smaller than $O(n)$ in OGD/SGD and $O(m)$ in RBCD. We show that the iteration complexity for ORBCD has the same order as OGD/SGD. In the stochastic setting, ORBCD is still suffered from the variance of stochastic gradient. To accelerate the convergence speed of ORBCD, we adopt the varaince reduction technique [93] to alleviate the effect of randomness. As expected, the linear convergence rate for ORBCD with variance reduction (ORBCDVD) is established for strongly convex functions for stochastic optimization. Moreover, ORBCDVD does not necessarily require to compute the full gradient at once which is necessary in SVRG and prox-SVRG. Instead, a block coordinate of full gradient is computed at each iteration and then stored for the next retrieval in ORBCDVD.

The rest of this chapter is organized as follows. In Section 2.2, we review the SGD and

RBCD. ORBCD and ORBCD with variance reduction are proposed in Section 2.3. The convergence results are given in Section 2.4.

2.2 Related Work

In this section, we briefly review the two types of low cost-per-iteration gradient descent (GD) methods, i.e., OGD/SGD and RBCD. Applying GD on (2.1), we have the following iterate:

$$\mathbf{x}^{t+1} = \text{argmin}_{\mathbf{x}} \ \langle \nabla f(\mathbf{x}^t), \mathbf{x} \rangle + g(\mathbf{x}) + \frac{\eta_t}{2} \|\mathbf{x} - \mathbf{x}^t\|_2^2 \,. \tag{2.2}$$

In some cases, e.g. $g(\mathbf{x})$ is ℓ_1 norm, (2.2) can have a closed-form solution.

2.2.1 Online and Stochastic Gradient Descent

In (2.2), it requires to compute the full gradient of m samples at each iteration, which could be computationally expensive if m is too large. Instead, OGD/SGD simply computes the gradient of one block samples.

In the online setting, at time $t+1$, OGD first presents a solution $\mathbf{x}^{t+1}$ by solving

$$\mathbf{x}^{t+1} = \text{argmin}_{\mathbf{x}} \ \langle \nabla f_t(\mathbf{x}^t), \mathbf{x} \rangle + g(\mathbf{x}) + \frac{\eta_t}{2} \|\mathbf{x} - \mathbf{x}^t\|_2^2 \,. \tag{2.3}$$

where f_t is given and assumed to be convex. Then a function f_{t+1} is revealed which incurs the loss $f_t(\mathbf{x}^t)$. The performance of OGD is measured by the regret bound, which is the discrepancy between the cumulative loss over T rounds and the best decision in hindsight,

$$R(T) = \sum_{t=1}^{T} [f_t(\mathbf{x}^t) + g(\mathbf{x}^t)] - [f_t(\mathbf{x}^*) + g(\mathbf{x}^*)] \,, \tag{2.4}$$

where $\mathbf{x}^*$ is the best result in hindsight. The regret bound of OGD is $O(\sqrt{T})$ when using decreasing step size $\eta_t = O(\frac{1}{\sqrt{t}})$. For strongly convex functions, the regret bound of OGD is $O(\log T)$ when using the step size $\eta_t = O(\frac{1}{t})$. Since f_t can be any convex function, OGD considers the worst case and thus the mentioned regret bounds are optimal.

In the stochastic setting, SGD first randomly picks up i_t-th block samples and then computes the gradient of the selected samples as follows:

$$\mathbf{x}^{t+1} = \text{argmin}_{\mathbf{x}} \ \langle \nabla f_{i_t}(\mathbf{x}^t), \mathbf{x} \rangle + g(\mathbf{x}) + \frac{\eta_t}{2} \|\mathbf{x} - \mathbf{x}^t\|_2^2 \,. \tag{2.5}$$

$\mathbf{x}^t$ depends on the observed realization of the random variable $\xi = \{i_1, \cdots, i_{t-1}\}$ or generally $\{\mathbf{x}^1, \cdots, \mathbf{x}^{t-1}\}$. Due to the effect of variance of stochastic gradient, SGD has to choose decreasing step size, i.e., $\eta_t = O(\frac{1}{\sqrt{t}})$, leading to slow convergence speed. For general convex functions, SGD converges at a rate of $O(\frac{1}{\sqrt{t}})$. For strongly convex functions, SGD converges at a rate of $O(\frac{1}{t})$. In contrast, GD converges linearly if functions are strongly convex.

To accelerate the SGD by reducing the variance of stochastic gradient, stochastic variance reduced gradient (SVRG) was proposed by [93]. [211] extends SVRG to composite functions (2.1), called prox-SVRG. SVRGs have two stages, i.e., outer stage and inner stage. The outer stage maintains an estimate $\tilde{\mathbf{x}}$ of the optimal point x^* and computes the full gradient of $\tilde{\mathbf{x}}$

$$\tilde{\mu} = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\tilde{\mathbf{x}}) = \nabla f(\tilde{\mathbf{x}}) \; . \tag{2.6}$$

After the inner stage is completed, the outer stage updates $\tilde{\mathbf{x}}$. At the inner stage, SVRG first randomly picks i_t-th sample, then modifies stochastis gradient by subtracting the difference between the full gradient and stochastic gradient at $\tilde{\mathbf{x}}$,

$$\mathbf{v}_t = \nabla f_{i_t}(\mathbf{x}^t) - \nabla f_{i_t}(\tilde{\mathbf{x}}) + \tilde{\mu} \; . \tag{2.7}$$

It can be shown that the expectation of $\mathbf{v}_t$ given $\mathbf{x}^{t-1}$ is the full gradient at $\mathbf{x}^t$, i.e., $\mathbb{E}\mathbf{v}_t = \nabla f(\mathbf{x}^t)$. Although $\mathbf{v}_t$ is also a stochastic gradient, the variance of stochastic gradient progressively decreases. Replacing $\nabla f_{i_t}(\mathbf{x}^t)$ by $\mathbf{v}_t$ in SGD step (2.5),

$$\mathbf{x}^{t+1} = \text{argmin}_{\mathbf{x}} \; \langle \mathbf{v}_t, \mathbf{x} \rangle + g(\mathbf{x}) + \frac{\eta}{2}\|\mathbf{x} - \mathbf{x}^t\|_2^2 \; . \tag{2.8}$$

By reduding the variance of stochastic gradient, $\mathbf{x}^t$ can converge to $\mathbf{x}^*$ at the same rate as GD, which has been proved in [93, 211]. For strongly convex functions, prox-SVRG [211] can converge linearly in expectation if $\eta > 4L$ and m satisfy the following condition:

$$\rho = \frac{\eta^2}{\gamma(\eta - 4L)m} + \frac{4L(m+1)}{(\eta - 4L)m} < 1 \; . \tag{2.9}$$

where L is the constant of Lipschitz continuous gradient. Note the step size is $1/\eta$ here.

2.2.2 Randomized Block Coordinate Descent

Assume $\mathbf{x}_j (1 \leq j \leq J)$ are non-overlapping blocks. At iteration t, RBCD [148, 160, 124] randomly picks j_t-th coordinate and solves the following problem:

$$\mathbf{x}_{j_t}^{t+1} = \text{argmin}_{\mathbf{x}_{j_t}} \; \langle \nabla_{j_t} f(\mathbf{x}^t), \mathbf{x}_{j_t} \rangle + g_{j_t}(\mathbf{x}_{j_t}) + \frac{\eta_t}{2}\|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2 \; . \tag{2.10}$$

Therefore, $\mathbf{x}^{t+1} = (\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{k \neq j_t}^t)$. $\mathbf{x}^t$ depends on the observed realization of the random variable

$$\xi = \{j_1, \cdots, j_{t-1}\} . \tag{2.11}$$

Setting the step size $\eta_t = L_{j_t}$ where L_{j_t} is the Lipshitz constant of j_t-th coordinate of the gradient $\nabla f(\mathbf{x}^t)$, the iteration complexity of RBCD is $O(\frac{1}{t})$. For strongly convex function, RBCD has a linear convergence rate. Therefore, RBCD converges at the same rate as GD, although the constant is J times larger [148, 160, 124].

2.3 Online Randomized Block Coordinate Descent

In this section, our goal is to combine OGD/SGD and RBCD together to solve problem (2.1). We call the algorithm online randomized block coordinate descent (ORBCD), which computes one block coordinate of the gradient of one block of samples at each iteration. ORBCD essentially performs RBCD in online and stochastic setting.

Let $\{\mathbf{x}_1, \cdots, \mathbf{x}_J\}, \mathbf{x}_j \in \mathbb{R}^{n_j \times 1}$ be J non-overlapping blocks of $\mathbf{x}$. Let $U_j \in \mathbb{R}^{n \times n_j}$ be n_j columns of an $n \times n$ permutation matrix $\mathbf{U}$, corresponding to j block coordinates in $\mathbf{x}$. For any partition of $\mathbf{x}$ and $\mathbf{U}$,

$$\mathbf{x} = \sum_{j=1}^{J} U_j \mathbf{x}_j \ , \mathbf{x}_j = U_j^T \mathbf{x} \ . \tag{2.12}$$

The j-th coordinate of gradient of f can be denoted as

$$\nabla_j f(\mathbf{x}) = U_j^T \nabla f(\mathbf{x}) \ . \tag{2.13}$$

Throughout the chapter, we assume that the minimum of problem (2.1) is attained. In addition, ORBCD needs the following assumption :

Assumption 1 *f_t or f_i has block-wise Lipschitz continuous gradient with constant L_j, e.g.,*

$$\|\nabla_j f_t(\mathbf{x} + U_j h_j) - \nabla_j f_t(\mathbf{x})\|_2 \leq L_j \|h_j\|_2 \leq L \|h_j\|_2 \ , \tag{2.14}$$

where $L = \max_j L_j$.

Assumption 2 *1. $\|\nabla f_t(\mathbf{x}^t)\|_2 \leq R_f$, or $\|\nabla f(\mathbf{x}^t)\|_2 \leq R_f$;*

2. $\mathbf{x}^t$ is assumed in a bounded set $\mathcal{X}$, i.e., $\sup_{\mathbf{x},\mathbf{y} \in \mathcal{X}} \|\mathbf{x} - \mathbf{y}\|_2 = D$.

While the Assumption 1 is used in RBCD, the Assumption 2 is used in OGD/SGD. We may assume the sum of two functions is strongly convex.

Assumption 3 *$f_t(\mathbf{x}) + g(\mathbf{x})$ or $f(\mathbf{x}) + g(\mathbf{x})$ is γ-strongly convex, e.g., we have*

$$f_t(\mathbf{x}) + g(\mathbf{x}) \geq f_t(\mathbf{y}) + g(\mathbf{y}) + \langle \nabla f_t(\mathbf{y}) + g'(\mathbf{y}), \mathbf{x} - \mathbf{x}^t \rangle + \frac{\gamma}{2}\|\mathbf{x} - \mathbf{y}\|_2^2 \,. \tag{2.15}$$

where $\gamma > 0$ and $g'(\mathbf{y})$ denotes the subgradient of g at $\mathbf{y}$.

2.3.1 ORBCD for Online Learning

In online setting, ORBCD considers the worst case and runs at rounds. At time t, given any function f_t which may be agnostic, ORBCD randomly chooses j_t-th block coordinate and presents the solution by solving the following problem:

$$\begin{aligned} \mathbf{x}_{j_t}^{t+1} &= \operatorname{argmin}_{\mathbf{x}_{j_t}} \ \langle \nabla_{j_t} f_t(\mathbf{x}^t), \mathbf{x}_{j_t} \rangle + g_{j_t}(\mathbf{x}_{j_t}) + \frac{\eta_t}{2}\|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2 \\ &= \text{Prox}_{g_{j_t}}(\mathbf{x}_{j_t} - \frac{1}{\eta_t}\nabla_{j_t} f_t(\mathbf{x}^t)) \,, \end{aligned} \tag{2.16}$$

where Prox denotes the proximal mapping. If f_t is a linear function, e.g., $f_t = l_t \mathbf{x}^t$, then $\nabla_{j_t} f_t(\mathbf{x}^t) = l_{j_t}$, so solving (2.16) is J times cheaper than OGD. Thus, $\mathbf{x}^{t+1} = (\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{k \neq j_t}^t)$, or

$$\mathbf{x}^{t+1} = \mathbf{x}^t + U_{j_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t) \,. \tag{2.17}$$

Then, ORBCD receives a loss function $f_{t+1}(\mathbf{x})$ which incurs the loss $f_{t+1}(\mathbf{x}^{t+1})$. The algorithm is summarized in Algorithm 1.

$\mathbf{x}^t$ is independent of j_t but depends on the sequence of observed realization of the random variable

$$\xi = \{j_1, \cdots, j_{t-1}\}. \tag{2.18}$$

Let $\mathbf{x}^*$ be the best solution in hindsight. The regret bound of ORBCD is defined as

$$R(T) = \sum_{t=1}^{T} \left\{ \mathbb{E}_{\xi}[f_t(\mathbf{x}^t) + g(\mathbf{x}^t)] - [f_t(\mathbf{x}^*) + g(\mathbf{x}^*)] \right\} \,. \tag{2.19}$$

By setting $\eta_t = \sqrt{t} + L$ where $L = \max_j L_j$, the regret bound of ORBCD is $O(\sqrt{T})$. For strongly convex functions, the regret bound of ORBCD is $O(\log T)$ by setting $\eta_t = \frac{\gamma t}{J} + L$.

Algorithm 1 Online Randomized Block Coordinate Descent for Online Learning

```
1: Initialization: $\mathbf{x}^1 = \mathbf{0}$
2: for $t = 1$ to $T$ do
3:     randomly pick up $j_t$ block coordinates
4:     $\mathbf{x}_{j_t}^{t+1} = \text{argmin}_{\mathbf{x}_{j_t} \in \mathcal{X}_j} \langle \nabla_{j_t} f_t(\mathbf{x}^t), \mathbf{x}_{j_t} \rangle + g_{j_t}(\mathbf{x}_{j_t}) + \frac{\eta_t}{2} \|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2$
5:     $\mathbf{x}^{t+1} = \mathbf{x}^t + U_{j_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)$
6:     receives the function $f_{t+1}(\mathbf{x}) + g(\mathbf{x})$ and incurs the loss $f_{t+1}(\mathbf{x}^{t+1}) + g(\mathbf{x}^{t+1})$
7: end for
```

2.3.2 ORBCD for Stochastic Optimization

In the stochastic setting, ORBCD first randomly picks up i_t-th block sample and then randomly chooses j_t-th block coordinate. The algorithm has the following iterate:

$$\begin{aligned} \mathbf{x}_{j_t}^{t+1} &= \text{argmin}_{\mathbf{x}_{j_t}} \langle \nabla_{j_t} f_{i_t}(\mathbf{x}^t), \mathbf{x}_{j_t} \rangle + g_{j_t}(\mathbf{x}_{j_t}) + \frac{\eta_t}{2} \|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2 \\ &= \text{Prox}_{g_{j_t}}(\mathbf{x}_{j_t} - \nabla_{j_t} f_{i_t}(\mathbf{x}^t)) \ . \end{aligned} \tag{2.20}$$

For high dimensional problem with non-overlapping composite regularizers, solving (2.20) is computationally cheaper than solving (2.5) in SGD. The algorithm of ORBCD in both settings is summarized in Algorithm 2.

$\mathbf{x}^{t+1}$ depends on (i_t, j_t), but j_t and i_t are independent. $\mathbf{x}^t$ is independent of (i_t, j_t) but depends on the observed realization of the random variables

$$\xi = \{(i_1, j_1), \cdots, (i_{t-1}, j_{t-1})\} \ . \tag{2.21}$$

The online-stochastic conversion rule [52, 53, 210] still holds here. The iteration complexity of ORBCD can be obtained by dividing the regret bounds in the online setting by T. Setting $\eta_t = \sqrt{t} + L$ where $L = \max_j L_j$, the iteration complexity of ORBCD is

$$\mathbb{E}_\xi[f(\bar{\mathbf{x}}^t) + g(\bar{\mathbf{x}}^t)] - [f(\mathbf{x}) + g(\mathbf{x})] \leq O(\frac{1}{\sqrt{T}}) \ . \tag{2.22}$$

For strongly convex functions, setting $\eta_t = \frac{\gamma t}{J} + L$,

$$\mathbb{E}_\xi[f(\bar{\mathbf{x}}^t) + g(\bar{\mathbf{x}}^t)] - [f(\mathbf{x}) + g(\mathbf{x})] \leq O(\frac{\log T}{T}) \ . \tag{2.23}$$

The iteration complexity of ORBCD match that of SGD. Simiarlar as SGD, the convergence speed of ORBCD is also slowed down by the variance of stochastic gradient.

Algorithm 2 Online Randomized Block Coordinate Descent for Stochastic Optimization

```
1: **Initialization:** $\mathbf{x}^1 = \mathbf{0}$
2: **for** $t = 1$ **to** $T$ **do**
3:     **randomly pick up** $i_t$ **block samples and** $j_t$ **block coordinates**
4:     $\mathbf{x}_{j_t}^{t+1} = \text{argmin}_{\mathbf{x}_{j_t} \in \mathcal{X}_j} \ \langle \nabla_{j_t} f_{i_t}(\mathbf{x}^t), \mathbf{x}_{j_t} \rangle + g_{j_t}(\mathbf{x}_{j_t}) + \frac{\eta_t}{2} \|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2$
5:     $\mathbf{x}^{t+1} = \mathbf{x}^t + U_{j_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)$
6: **end for**
```

Algorithm 3 Online Randomized Block Coordinate Descent with Variance Reduction

[tb]

```
1: **Initialization:** $\mathbf{x}^1 = \mathbf{0}$
2: **for** $t = 2$ to $T$ **do**
3:     $\mathbf{x}_0 = \tilde{\mathbf{x}} = \mathbf{x}^t$.
4:     **for** $k = 0$ **to** $m - 1$ **do**
5:         randomly pick up $i_k$ block samples
6:         randomly pick up $j_k$ block coordinates
7:         $\mathbf{v}_{j_k}^{i_k} = \nabla_{j_k} f_{i_k}(\mathbf{x}^k) - \nabla_{j_k} f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}_{j_k}$ where $\tilde{\mu}_{j_k} = \nabla_{j_k} f(\tilde{\mathbf{x}})$
8:         $\mathbf{x}_{j_k}^k = \text{argmin}_{\mathbf{x}_{j_k}} \ \langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k} \rangle + g_{j_k}(\mathbf{x}_{j_k}) + \frac{\eta_k}{2} \|\mathbf{x}_{j_k} - \mathbf{x}_{j_k}^k\|_2^2$
9:         $\mathbf{x}^{k+1} = \mathbf{x}^k + U_{j_k}(\mathbf{x}_{j_j}^{k+1} - \mathbf{x}_{j_k}^k)$
10:     **end for**
11:     $\mathbf{x}^{t+1} = \mathbf{x}^m$ or $\frac{1}{m} \sum_{k=1}^m \mathbf{x}^k$
12: **end for**
```

2.3.3 ORBCD with variance reduction

In the stochastic setting, we apply the variance reduction technique [211, 93] to accelerate the rate of convergence of ORBCD, abbreviated as ORBCDVD. As SVRG and prox-SVRG, ORBCDVD consists of two stages. At time $t + 1$, the outer stage maintains an estimate $\tilde{\mathbf{x}} = \mathbf{x}^t$ of the optimal $\mathbf{x}^*$ and updates $\tilde{\mathbf{x}}$ every $m + 1$ iterations. The inner stage takes m iterations which is indexed by $k = 0, \cdots, m - 1$. At the k-th iteration, ORBCDVD randomly picks i_k-th sample and j_k-th coordinate and compute

$$\mathbf{v}_{j_k}^{i_k} = \nabla_{j_k} f_{i_k}(\mathbf{x}^k) - \nabla_{j_k} f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}_{j_k} \ , \tag{2.24}$$

where

$$\tilde{\mu}_{j_k} = \frac{1}{n}\sum_{i=1}^{n} \nabla_{j_k} f_i(\tilde{\mathbf{x}}) = \nabla_{j_k} f(\tilde{\mathbf{x}}) \ . \tag{2.25}$$

$\mathbf{v}_{j_t}^{i_t}$ depends on (i_t, j_t), and i_t and j_t are independent. Conditioned on $\mathbf{x}^k$, taking expectation over i_k, j_k gives

$$\begin{aligned}\mathbb{E}\mathbf{v}_{j_k}^{i_k} &= \mathbb{E}_{i_k}\mathbb{E}_{j_k}[\nabla_{j_k} f_{i_k}(\mathbf{x}^k) - \nabla_{j_k} f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}_{j_k}] \\ &= \frac{1}{J}\mathbb{E}_{i_k}[\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}] \\ &= \frac{1}{J}\nabla f(\mathbf{x}^k) \ . \end{aligned} \tag{2.26}$$

Although $\mathbf{v}_{j_k}^{i_k}$ is stochastic gradient, the variance $\mathbb{E}\|\mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2$ decreases progressively and is smaller than $\mathbb{E}\|\nabla f_{i_t}(\mathbf{x}^t) - \nabla f(\mathbf{x}^t)\|_2^2$. Using the variance reduced gradient $\mathbf{v}_{j_k}^{i_k}$, ORBCD then performs RBCD as follows:

$$\mathbf{x}_{j_k}^{k+1} = \text{argmin}_{\mathbf{x}_{j_k}} \ \langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k}\rangle + g_{j_k}(\mathbf{x}_{j_k}) + \frac{\eta}{2}\|\mathbf{x}_{j_k} - \mathbf{x}_{j_k}^k\|_2^2 \ . \tag{2.27}$$

After m iterations, the outer stage updates $\mathbf{x}^{t+1}$ which is either $\mathbf{x}^m$ or $\frac{1}{m}\sum_{k=1}^m \mathbf{x}^k$. The algorithm is summarized in Algorithm 3. At the outer stage, ORBCDVD does not necessarily require to compute the full gradient at once. If the computation of full gradient requires substantial computational eorts, SVRG has to stop and complete the full gradient step before making progress. In contrast, $\tilde{\mu}$ can be partially computed at each iteration and then stored for the next retrieval in ORBCDVD.

Assume $\eta > 2L$ and m satisfy the following condition:

$$\rho = \frac{L}{\eta - 2L} + \frac{(\eta - L)J}{(\eta - 2L)m} - \frac{1}{m} + \frac{\eta(\eta - L)J}{(\eta - 2L)m\gamma} < 1 \ , \tag{2.28}$$

Then $h(\mathbf{x})$ converges linearly in expectation, i.e.,

$$\mathbb{E}_\xi[f(\mathbf{x}^t) + g(\mathbf{x}^t) - (f(\mathbf{x}^*) + g(\mathbf{x}^*)] \leq O(\rho^t) \ . \tag{2.29}$$

Setting $\eta = 4L$ in (2.28) yields

$$\rho = \frac{1}{2} + \frac{3J}{2m} - \frac{1}{m} + \frac{6JL}{m\gamma} \leq \frac{1}{2} + \frac{3J}{2m}(1 + \frac{4L}{\gamma}) \ . \tag{2.30}$$

Setting $m = 18JL/\gamma$, then

$$\rho \leq \frac{1}{2} + \frac{1}{12}(\frac{\gamma}{L} + 4) \approx \frac{11}{12} \ . \tag{2.31}$$

where we assume $\gamma/L \approx 1$ for simplicity.

2.4 The Rate of Convergence

The following lemma is a key building block of the proof of the convergence of ORBCD in both online and stochastic setting.

Lemma 1 *Let the Assumption 1 and 2 hold. Let* $\mathbf{x}^t$ *be the sequences generated by ORBCD.* j_t *is sampled randomly and uniformly from* $\{1, \cdots, J\}$*. We have*

$$\langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t_{j_t}), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t} \rangle \leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{R_f^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - g(\mathbf{x}^{t+1}) . \tag{2.32}$$

where $L = \max_j L_j$*.*

Proof: The optimality condition is

$$\langle \nabla_{j_t} f_t(\mathbf{x}^t) + \eta_t(\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}) + g'_{j_t}(\mathbf{x}^{t+1}_{j_t}), \mathbf{x}^{t+1}_{j_t} - \mathbf{x}_{j_t} \rangle \leq 0 . \tag{2.33}$$

Rearranging the terms yields

$$\begin{aligned} &\langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^{t+1}_{j_t}), \mathbf{x}^{t+1}_{j_t} - \mathbf{x}_{j_t} \rangle \\ &\leq -\eta_t \langle \mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}, \mathbf{x}^{t+1}_{j_t} - \mathbf{x}_{j_t} \rangle \\ &\leq \frac{\eta_t}{2}(\|\mathbf{x}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2 - \|\mathbf{x}_{j_t} - \mathbf{x}^{t+1}_{j_t}\|_2^2 - \|\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2) \\ &= \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2 - \|\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2) , \end{aligned} \tag{2.34}$$

where the last equality uses $\mathbf{x}^{t+1} = (\mathbf{x}^{t+1}_{j_t}, \mathbf{x}^t_{k \neq j_t})$. By the smoothness of f_t, we have

$$f_t(\mathbf{x}^{t+1}) \leq f_t(\mathbf{x}^t) + \langle \nabla_j f_t(\mathbf{x}^t), \mathbf{x}^{t+1}_j - \mathbf{x}^t_j \rangle + \frac{L_j}{2}\|\mathbf{x}^{t+1}_j - \mathbf{x}^t_j\|_2^2 . \tag{2.35}$$

Since $\mathbf{x}^{t+1} - \mathbf{x}^t = U_{j_t}(\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t})$, we have

$$\begin{aligned} &f_t(\mathbf{x}^{t+1}) + g(\mathbf{x}^{t+1}) - [f_t(\mathbf{x}^t) + g(\mathbf{x}^t)] \\ &\leq \langle \nabla_{j_t} f_t(\mathbf{x}^t), \mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t} \rangle + \frac{L_{j_t}}{2}\|\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2 + g_{j_t}(\mathbf{x}^{t+1}_{j_t}) - g_{j_t}(\mathbf{x}_{j_t}) + g_{j_t}(\mathbf{x}^t_{j_t}) - g_{j_t}(\mathbf{x}_{j_t}) \\ &\leq \langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^{t+1}_{j_t}), \mathbf{x}^{t+1}_{j_t} - \mathbf{x}_{j_t} \rangle + \frac{L_{j_t}}{2}\|\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2 - \langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t_{j_t}), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t} \rangle \\ &\leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{L_{j_t} - \eta_t}{2}\|\mathbf{x}^{t+1}_{j_t} - \mathbf{x}^t_{j_t}\|_2^2 - \langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t_{j_t}), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t} \rangle . \end{aligned} \tag{2.36}$$

Rearranging the terms yields

$$\langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t), \mathbf{x}_{j_t}^t - \mathbf{x}_{j_t} \rangle \leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{L_{j_t} - \eta_t}{2} \|\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t\|_2^2$$
$$+ f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}^{t+1}) + g(\mathbf{x}^{t+1})] . \tag{2.37}$$

The convexity of f_t gives

$$\begin{aligned} f_t(\mathbf{x}^t) - f_t(\mathbf{x}^{t+1}) &\leq \langle \nabla f_t(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}^{t+1} \rangle \\ &= \langle \nabla_{j_t} f_t(\mathbf{x}^t), \mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^{t+1} \rangle \\ &\leq \frac{1}{2\alpha} \|\nabla_{j_t} f_t(\mathbf{x}^t)\|_2^2 + \frac{\alpha}{2} \|\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^{t+1}\|_2^2 . \end{aligned} \tag{2.38}$$

where the equality uses $\mathbf{x}^{t+1} = (\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{k \neq j_t}^t)$. Plugging into (2.37), we have

$$\begin{aligned} &\langle \nabla_{j_t} f_t(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}_{j_t}^t), \mathbf{x}_{j_t}^t - \mathbf{x}_{j_t} \rangle \\ &\leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{L_{j_t} - \eta_t}{2} \|\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t\|_2^2 \\ &+ \langle \nabla_{j_t} f_t(\mathbf{x}^t), \mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^{t+1} \rangle + g(\mathbf{x}^t) - g(\mathbf{x}^{t+1}) \\ &\leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{L_{j_t} - \eta_t}{2} \|\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t\|_2^2 \\ &+ \frac{\alpha}{2} \|\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^{t+1}\|_2^2 + \frac{1}{2\alpha} \|\nabla_{j_t} f_t(\mathbf{x}^t)\|_2^2 . \end{aligned} \tag{2.39}$$

Let $L = \max_j L_j$. Setting $\alpha = \eta_t - L$ where $\eta_t > L$ completes the proof. ■

This lemma is also a key building block in the proof of iteration complexity of GD, OGD/SGD and RBCD. In GD, by setting $\eta_t = L$, the iteration complexity of GD can be established. In RBCD, by simply setting $\eta_t = L_{j_t}$, the iteration complexity of RBCD can be established.

2.4.1 Online Optimization

Note $\mathbf{x}^t$ depends on the sequence of observed realization of the random variable $\xi = \{j_1, \cdots, j_{t-1}\}$. The following theorem establishes the regret bound of ORBCD.

Theorem 1 *Let $\eta_t = \sqrt{t} + L$ in the ORBCD and the Assumption 1 and 2 hold. j_t is sampled randomly and uniformly from $\{1, \cdots, J\}$. The regret bound $R(T)$ of ORBCD is*

$$R(T) \leq J(\frac{\sqrt{T} + L}{2} D^2 + \sqrt{T} R^2 + g(\mathbf{x}^1) - g(\mathbf{x}^*)) . \tag{2.40}$$

Proof: In (2.32), conditioned on $\mathbf{x}^t$, take expectation over j_t, we have

$$\frac{1}{J}\langle \nabla f_t(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle \leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - \mathbb{E}g(\mathbf{x}^{t+1}) \; . \tag{2.41}$$

Using the convexity, we have

$$f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})] \leq \langle \nabla f_t(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle \; . \tag{2.42}$$

Together with (2.41), we have

$$f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})] \leq J\left\{\frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - \mathbb{E}g(\mathbf{x}^{t+1})\right\} \; . \tag{2.43}$$

Taking expectation over ξ on both sides, we have

$$\mathbb{E}_\xi\left[f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})]\right] \leq J\left\{\frac{\eta_t}{2}(\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{R^2}{2(\eta_t - L)} + \mathbb{E}_\xi g(\mathbf{x}^t) - \mathbb{E}_\xi g(\mathbf{x}^{t+1})\right\} \; . \tag{2.44}$$

Summing over t and setting $\eta_t = \sqrt{t} + L$, we obtain the regret bound

$$\begin{aligned} R(T) &= \sum_{t=1}^{T}\left\{\mathbb{E}_\xi[f_t(\mathbf{x}^t) + g(\mathbf{x}^t)] - [f_t(\mathbf{x}) + g(\mathbf{x})]\right\} \\ &\leq J\left\{-\frac{\eta_T}{2}\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^{T+1}\|_2^2 + \sum_{t=1}^{T}(\eta_t - \eta_{t-1})\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^t\|_2^2 \right.\\ &\left. + \sum_{t=1}^{T}\frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^1) - \mathbb{E}_\xi g(\mathbf{x}^{T+1})\right\} \\ &\leq J\left\{\frac{\eta_T}{2}D^2 + \sum_{t=1}^{T}\frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^1) - g(\mathbf{x}^*)\right\} \\ &\leq J\left\{\frac{\sqrt{T} + L}{2}D^2 + \sum_{t=1}^{T}\frac{R^2}{2\sqrt{t}} + g(\mathbf{x}^1) - g(\mathbf{x}^*)\right\} \\ &\leq J(\frac{\sqrt{T} + L}{2}D^2 + \sqrt{T}R^2 + g(\mathbf{x}^1) - g(\mathbf{x}^*)) \; , \end{aligned} \tag{2.45}$$

which completes the proof. ■

If one of the functions is strongly convex, ORBCD can achieve a $\log(T)$ regret bound, which is established in the following theorem.

Theorem 2 *Let the Assumption 1-3 hold and $\eta_t = \frac{\gamma t}{J} + L$ in ORBCD. j_t is sampled randomly and uniformly from $\{1, \cdots, J\}$. The regret bound $R(T)$ of ORBCD is*

$$R(T) \leq J^2 R^2 \log(T) + J(g(\mathbf{x}^1) - g(\mathbf{x}^*)) . \tag{2.46}$$

Proof: Using the strong convexity of $f_t + g$ in (2.15), we have

$$f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})] \leq \langle \nabla f_t(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x} \rangle - \frac{\gamma}{2}\|\mathbf{x} - \mathbf{x}^t\|_2^2 . \tag{2.47}$$

Together with (2.41), we have

$$\begin{aligned} f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})] \leq& \frac{J\eta_t - \gamma}{2}\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \frac{J\eta_t}{2}\mathbb{E}\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) \\ &+ \frac{JR^2}{2(\eta_t - L)} + J[g(\mathbf{x}^t) - \mathbb{E}g(\mathbf{x}^{t+1})] . \end{aligned} \tag{2.48}$$

Taking expectation over ξ on both sides, we have

$$\begin{aligned} \mathbb{E}_\xi\left[f_t(\mathbf{x}^t) + g(\mathbf{x}^t) - [f_t(\mathbf{x}) + g(\mathbf{x})]\right] \leq& \frac{J\eta_t - \gamma}{2}\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \frac{J\eta_t}{2}\mathbb{E}_\xi[\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2]) \\ &+ \frac{JR^2}{2(\eta_t - L)} + J[\mathbb{E}_\xi g(\mathbf{x}^t) - \mathbb{E}_\xi g(\mathbf{x}^{t+1})] . \end{aligned} \tag{2.49}$$

Summing over t and setting $\eta_t = \frac{\gamma t}{J} + L$, we obtain the regret bound

$$\begin{aligned} R(T) =& \sum_{t=1}^{T}\left\{\mathbb{E}_\xi[f_t(\mathbf{x}^t) + g(\mathbf{x}^t)] - [f_t(\mathbf{x}) + g(\mathbf{x})]\right\} \\ \leq& -\frac{J\eta_T}{2}\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^{T+1}\|_2^2 + \sum_{t=1}^{T}\frac{J\eta_t - \gamma - J\eta_{t-1}}{2}\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^t\|_2^2 \\ &+ \sum_{t=1}^{T}\frac{JR^2}{2(\eta_t - L)} + J(g(\mathbf{x}^1) - \mathbb{E}_\xi g(\mathbf{x}^{T+1})) \\ \leq& \sum_{t=1}^{T}\frac{J^2R^2}{2\gamma t} + J(g(\mathbf{x}^1) - g(\mathbf{x}^*)) \\ \leq& J^2R^2\log(T) + J(g(\mathbf{x}^1) - g(\mathbf{x}^*)) , \end{aligned} \tag{2.50}$$

which completes the proof. ■

In general, ORBCD can achieve the same order of regret bound as OGD and other first-order online optimization methods, although the constant could be J times larger.

2.4.2 Stochastic Optimization

In the stochastic setting, ORBCD first randomly chooses the i_t-th block sample and the j_t-th block coordinate. j_t and i_t are independent. $\mathbf{x}^t$ depends on the observed realization of the random variables $\xi = \{(i_1, j_1), \cdots, (i_{t-1}, j_{t-1})\}$. The following theorem establishes the iteration complexity of ORBCD for general convex functions.

Theorem 3 *Let $\eta_t = \sqrt{t} + L$ and $\bar{\mathbf{x}}^T = \frac{1}{T}\sum_{t=1}^T \mathbf{x}^t$ in the ORBCD. i_t, j_t are sampled randomly and uniformly from $\{1, \cdots, I\}$ and $\{1, \cdots, J\}$ respectively. The iteration complexity of ORBCD is*

$$\mathbb{E}_\xi[f(\bar{\mathbf{x}}^t) + g(\bar{\mathbf{x}}^t)] - [f(\mathbf{x}) + g(\mathbf{x})] \leq \frac{J(\frac{\sqrt{T}+L}{2}D^2 + \sqrt{T}R^2 + g(\mathbf{x}^1) - g(\mathbf{x}^*))}{T} . \qquad (2.51)$$

Proof: In the stochastic setting, let f_t be f_{i_t} in (2.32), we have

$$\begin{aligned}\langle \nabla_{j_t} f_{i_t}(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t}\rangle &\leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) \\ &\quad + \frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - g(\mathbf{x}^{t+1}) . \qquad (2.52)\end{aligned}$$

Note i_t, j_t are independent of $\mathbf{x}^t$. Conditioned on $\mathbf{x}^t$, taking expectation over i_t and j_t, the RHS is

$$\begin{aligned}&\mathbb{E}\langle \nabla_{j_t} f_{i_t}(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t}\rangle = \mathbb{E}_{i_t}[\mathbb{E}_{j_t}[\langle \nabla_{j_t} f_{i_t}(\mathbf{x}^t) + g'_{j_t}(\mathbf{x}^t), \mathbf{x}^t_{j_t} - \mathbf{x}_{j_t}\rangle]] \\ &= \frac{1}{J}\mathbb{E}_{i_t}[\langle \nabla f_{i_t}(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle + \langle g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle] \\ &= \frac{1}{J}\langle \nabla f(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle . \qquad (2.53)\end{aligned}$$

Plugging back into (2.52), we have

$$\begin{aligned}&\frac{1}{J}\langle \nabla f(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x}\rangle \\ &\leq \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) + \frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - \mathbb{E}g(\mathbf{x}^{t+1}) . \qquad (2.54)\end{aligned}$$

Using the convexity of $f + g$, we have

$$f(\mathbf{x}^t) + g(\mathbf{x}^t) - [f(\mathbf{x}) + g(\mathbf{x})] \leq \langle \nabla f(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x} \rangle \, . \tag{2.55}$$

Together with (2.54), we have

$$\begin{aligned} f(\mathbf{x}^t) + g(\mathbf{x}^t) - [f(\mathbf{x}) + g(\mathbf{x})] \leq J \bigg\{ & \frac{\eta_t}{2}(\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2) \\ & + \frac{R^2}{2(\eta_t - L)} + g(\mathbf{x}^t) - \mathbb{E}g(\mathbf{x}^{t+1}) \bigg\} \, . \end{aligned} \tag{2.56}$$

Taking expectation over ξ on both sides, we have

$$\begin{aligned} \mathbb{E}_\xi \left[f(\mathbf{x}^t) + g(\mathbf{x}^t) \right] - [f(\mathbf{x}) + g(\mathbf{x})] \leq J \bigg\{ & \frac{\eta_t}{2}(\mathbb{E}_\xi\|\mathbf{x} - \mathbf{x}^t\|_2^2 - \mathbb{E}_\xi[\|\mathbf{x} - \mathbf{x}^{t+1}\|_2^2]) \\ & + \frac{R^2}{2(\eta_t - L)} + \mathbb{E}_\xi g(\mathbf{x}^t) - \mathbb{E}_\xi g(\mathbf{x}^{t+1}) \bigg\} \, . \end{aligned} \tag{2.57}$$

Summing over t and setting $\eta_t = \sqrt{t} + L$, following similar derivation in (2.45), we have

$$\sum_{t=1}^{T} \left\{ \mathbb{E}_\xi [f(\mathbf{x}^t) + g(\mathbf{x}^t)] - [f(\mathbf{x}) + g(\mathbf{x})] \right\} \leq J(\frac{\sqrt{T} + L}{2} D^2 + \sqrt{T} R^2 + g(\mathbf{x}^1) - g(\mathbf{x}^*)) \, . \tag{2.58}$$

Dividing both sides by T, using the Jensen's inequality and denoting $\bar{\mathbf{x}}^T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}^t$ complete the proof. ■

For strongly convex functions, we have the following results.

Theorem 4 *For strongly convex function, setting $\eta_t = \frac{\gamma t}{J} + L$ in the ORBCD. i_t, j_t are sampled randomly and uniformly from $\{1, \cdots, I\}$ and $\{1, \cdots, J\}$ respectively. Let $\bar{\mathbf{x}}^T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}^t$. The iteration complexity of ORBCD is*

$$\mathbb{E}_\xi [f(\bar{\mathbf{x}}^T) + g(\bar{\mathbf{x}}^T)] - [f(\mathbf{x}) + g(\mathbf{x})] \leq \frac{J^2 R^2 \log(T) + J(g(\mathbf{x}^1) - g(\mathbf{x}^*))}{T} \, . \tag{2.59}$$

Proof: If $f + g$ is strongly convex, we have

$$f(\mathbf{x}^t) + g(\mathbf{x}^t) - [f(\mathbf{x}) + g(\mathbf{x})] \leq \langle \nabla f(\mathbf{x}^t) + g'(\mathbf{x}^t), \mathbf{x}^t - \mathbf{x} \rangle - \frac{\gamma}{2}\|\mathbf{x} - \mathbf{x}^t\|_2^2 \, . \tag{2.60}$$

Plugging back into (2.54), following similar derivation in Theorem 2 and Theorem 3 complete the proof. ■

2.4.3 ORBCD with Variance Reduction

According to the Theorem 2.1.5 in [145], the block-wise Lipschitz gradient in Assumption 1 can also be rewritten as follows:

$$f_i(\mathbf{x}) \le f_i(\mathbf{y}) + \langle \nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{y}), \mathbf{x}_j - \mathbf{y}_j \rangle + \frac{L}{2}\|\mathbf{x}_j - \mathbf{y}_j\|_2^2 \,, \tag{2.61}$$

$$\|\nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{y})\|_2^2 \le L\langle \nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{y}), \mathbf{x}_j - \mathbf{y}_j \rangle \,. \tag{2.62}$$

Let $\mathbf{x}^*$ be an optimal solution. Define an upper bound of $f(\mathbf{x}) + g(\mathbf{x}) - (f(\mathbf{x}^*) + g(\mathbf{x}^*))$ as

$$h(\mathbf{x}, \mathbf{x}^*) = \langle \nabla f(\mathbf{x}), \mathbf{x} - \mathbf{x}^* \rangle + g(\mathbf{x}) - g(\mathbf{x}^*) \,. \tag{2.63}$$

If $f(\mathbf{x}) + g(\mathbf{x})$ is strongly convex, we have

$$h(\mathbf{x}, \mathbf{x}^*) \ge f(\mathbf{x}) - f(\mathbf{x}^*) + g(\mathbf{x}) - g(\mathbf{x}^*) \ge \frac{\gamma}{2}\|\mathbf{x} - \mathbf{x}^*\|_2^2 \,. \tag{2.64}$$

Lemma 2 *Let $\mathbf{x}^*$ be an optimal solution and the Assumption 1 hold, we have*

$$\frac{1}{I}\sum_{i=1}^{I} \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|_2^2 \le L h(\mathbf{x}, \mathbf{x}^*) \,. \tag{2.65}$$

where h is defined in (5.23).

Proof: Since the Assumption 1 hold, we have

$$\begin{aligned} \frac{1}{I}\sum_{i=1}^{I} \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|_2^2 &= \frac{1}{I}\sum_{i=1}^{I}\sum_{j=1}^{J} \|\nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{x}^*)\|_2^2 \\ &\le \frac{1}{I}\sum_{i=1}^{I}\sum_{j=1}^{J} L\langle \nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{x}^*), \mathbf{x}_j - \mathbf{x}_j^* \rangle \\ &= L[\langle \nabla f(\mathbf{x}), \mathbf{x} - \mathbf{x}^* \rangle + \langle \nabla f(\mathbf{x}^*), \mathbf{x}^* - \mathbf{x} \rangle] \,, \end{aligned} \tag{2.66}$$

where the inequality uses (2.62). For an optimal solution $\mathbf{x}^*$, $g'(\mathbf{x}^*) + \nabla f(\mathbf{x}^*) = 0$ where $g'(\mathbf{x}^*)$ is the subgradient of g at $\mathbf{x}^*$. The second term in (2.66) can be rewritten as

$$\langle \nabla f(\mathbf{x}^*), \mathbf{x}^* - \mathbf{x} \rangle = -\langle g'(\mathbf{x}^*), \mathbf{x}^* - \mathbf{x} \rangle = g(\mathbf{x}) - g(\mathbf{x}^*) \,. \tag{2.67}$$

Plugging into (2.66) and using (5.23) complete the proof. ■

Lemma 3 *Let $\mathbf{x}^*$ be an optimal solution and the Assumption 1 hold, we have*

$$h(\mathbf{x}, \mathbf{x}^*) \leq L\|\mathbf{x} - \mathbf{x}^*\|_2^2 \,. \tag{2.68}$$

where h is defined in (5.23).

Proof: Since the Assumption 1 hold, we have

$$\begin{aligned} \langle \nabla f(\mathbf{x}) - \nabla f(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle &= \frac{1}{I}\sum_{i=1}^{I}\sum_{j=1}^{J}\langle \nabla_j f_i(\mathbf{x}) - \nabla_j f_i(\mathbf{x}^*), \mathbf{x}_j - \mathbf{x}_j^* \rangle \\ \leq \frac{1}{I}\sum_{i=1}^{I}\sum_{j=1}^{J} L\|\mathbf{x}_j - \mathbf{x}_j^*\|_2^2 &= L\|\mathbf{x} - \mathbf{x}^*\|_2^2 \,. \end{aligned} \tag{2.69}$$

As shown in Lemma 2,

$$\langle \nabla f(\mathbf{x}) - \nabla f(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle = h(\mathbf{x}, \mathbf{x}^*) \,, \tag{2.70}$$

which completes the proof. ■

Lemma 4 *Let $\mathbf{v}_{j_k}^{i_k}$ and $\mathbf{x}_{j_k}^{k+1}$ be generated by (2.24)-(2.27). Conditioned on $\mathbf{x}^k$, we have*

$$\mathbb{E}\|\mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2 \leq \frac{2L}{J}[h(\mathbf{x}^k, \mathbf{x}^*) + h(\tilde{\mathbf{x}}, \mathbf{x}^*)] \,. \tag{2.71}$$

Proof: Conditioned on $\mathbf{x}^k$, we have

$$\mathbb{E}_{i_k}[\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}] = \frac{1}{I}\sum_{i=1}^{I}[\nabla f_i(\mathbf{x}^k) - \nabla f_i(\tilde{\mathbf{x}}) + \tilde{\mu}] = \nabla f(\mathbf{x}^k) \,. \tag{2.72}$$

Note $\mathbf{x}^k$ is independent of i_k, j_k. i_k and j_k are independent. Conditioned on $\mathbf{x}^k$, taking expectation over i_k, j_k and using (2.24) give

$$\begin{aligned} &\mathbb{E}\|\mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2 = \mathbb{E}_{i_k}[\mathbb{E}_{j_k}\|\mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2] \\ &= \mathbb{E}_{i_k}[\mathbb{E}_{j_k}\|\nabla_{j_k} f_{i_k}(\mathbf{x}^k) - \nabla_{j_k} f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu}_{j_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2] \\ &= \frac{1}{J}\mathbb{E}_{i_k}\|\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \tilde{\mu} - \nabla f(\mathbf{x}^k)\|_2^2 \\ &\leq \frac{1}{J}\mathbb{E}_{i_k}\|\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}})\|_2^2 \\ &\leq \frac{2}{J}\mathbb{E}_{i_k}\|\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\mathbf{x}^*)\|_2^2 + \frac{2}{J}\mathbb{E}_{i_k}\|\nabla f_{i_k}(\tilde{\mathbf{x}}) - \nabla f_{i_k}(\mathbf{x}^*)\|_2^2 \end{aligned}$$

$$= \frac{2}{IJ}\sum_{i=1}^{I}\|\nabla f_i(\mathbf{x}^k) - \nabla f_i(\mathbf{x}^*)\|_2^2 + \frac{2}{IJ}\sum_{i=1}^{I}\|\nabla f_i(\tilde{\mathbf{x}}) - \nabla f_i(\mathbf{x}^*)\|_2^2$$
$$\leq \frac{2L}{J}[h(\mathbf{x}^k, \mathbf{x}^*) + h(\tilde{\mathbf{x}}, \mathbf{x}^*)] \ . \tag{2.73}$$

The first inequality uses the fact $\mathbb{E}\|\zeta - \mathbb{E}\zeta\|_2^2 \leq \mathbb{E}\|\zeta\|_2^2$ given a random variable ζ, the second inequality uses $\|\mathbf{a}+\mathbf{b}\|_2^2 \leq 2\|\mathbf{a}\|_2^2 + 2\|\mathbf{b}\|_2^2$, and the last inequality uses Lemma 2. ■

Lemma 5 *Under Assumption 1,* $f(\mathbf{x}) = \frac{1}{I}\sum_{i=1}^{I} f_i(\mathbf{x})$ *has block-wise Lipschitz continuous gradient with constant L, i.e.,*

$$\|\nabla_j f(\mathbf{x} + U_j h_j) - \nabla_j f(\mathbf{x})\|_2 \leq L\|h_j\|_2 \ . \tag{2.74}$$

Proof: Using the fact that $f(\mathbf{x}) = \frac{1}{I}\sum_{i=1}^{I} f_i(\mathbf{x})$, we have

$$\|\nabla_j f(\mathbf{x} + U_j h_j) - \nabla_j f(\mathbf{x})\|_2 = \|\frac{1}{I}\sum_{i=1}^{I}[\nabla_j f_i(\mathbf{x} + U_j h_j) - \nabla_j f_i(\mathbf{x})]\|_2$$
$$\leq \frac{1}{I}\sum_{i=1}^{I}\|\nabla_j f_i(\mathbf{x} + U_j h_j) - \nabla_j f_i(\mathbf{x})\|_2$$
$$\leq L\|h_j\|_2 \ , \tag{2.75}$$

where the first inequality uses the Jensen's inequality and the second inequality uses the Assumption 1. ■

Now, we are ready to establish the linear convergence rate of ORBCD with variance reduction for strongly convex functions.

Theorem 5 *Let* $\mathbf{x}^t$ *be generated by ORBCD with variance reduction (2.25)-(2.27).* j_k *is sampled randomly and uniformly from* $\{1, \cdots, J\}$*. Assume* $\eta > 2L$ *and* m *satisfy the following condition:*

$$\rho = \frac{L}{\eta - 2L} + \frac{(\eta - L)J}{(\eta - 2L)m} - \frac{1}{m} + \frac{\eta(\eta - L)J}{(\eta - 2L)m\gamma} < 1 \ , \tag{2.76}$$

Then ORBCDVD converges linearly in expectation, i.e.,

$$\mathbb{E}_\xi[f(\mathbf{x}^t) + g(\mathbf{x}^t) - (f(\mathbf{x}^*) + g(\mathbf{x}^*)] \leq \rho^t[\mathbb{E}_\xi h(\mathbf{x}^1, \mathbf{x}^*)] \ . \tag{2.77}$$

where h *is defined in (5.23).*

Proof: The optimality condition of (2.27) is

$$\langle \mathbf{v}_{j_k}^{i_k} + \eta(\mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}) + g'_{j_k}(\mathbf{x}_{j_k}^{k+1}), \mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k} \rangle \le 0 \,. \tag{2.78}$$

Rearranging the terms yields

$$\begin{aligned}&\langle \mathbf{v}_{j_k}^{i_k} + g'_{j_k}(\mathbf{x}_{j_k}^{k+1}), \mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k} \rangle \le -\eta \langle \mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}, \mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k} \rangle \\ &\le \frac{\eta}{2}(\|\mathbf{x}_{j_k} - \mathbf{x}_{j_k}^{k}\|_2^2 - \|\mathbf{x}_{j_k} - \mathbf{x}_{j_k}^{k+1}\|_2^2 - \|\mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}\|_2^2) \\ &= \frac{\eta}{2}(\|\mathbf{x} - \mathbf{x}^{k}\|_2^2 - \|\mathbf{x} - \mathbf{x}^{k+1}\|_2^2 - \|\mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}\|_2^2) \,,\end{aligned} \tag{2.79}$$

where the last equality uses $\mathbf{x}^{k+1} = (\mathbf{x}_{j_k}^{k+1}, \mathbf{x}_{k \neq j_k}^{t})$. Using the convecxity of g_j and the fact that $g(\mathbf{x}^k) - g(\mathbf{x}^{k+1}) = g_{j_k}(\mathbf{x}^k) - g_{j_k}(\mathbf{x}^{k+1})$, we have

$$\begin{aligned}&\langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k} \rangle + g_{j_k}(\mathbf{x}^k) - g_{j_k}(\mathbf{x}) \le \langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{k+1} \rangle + g(\mathbf{x}^k) - g(\mathbf{x}^{k+1}) \\ &+ \frac{\eta}{2}(\|\mathbf{x} - \mathbf{x}^{k}\|_2^2 - \|\mathbf{x} - \mathbf{x}^{k+1}\|_2^2 - \|\mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}\|_2^2) \,.\end{aligned} \tag{2.80}$$

According to Lemma 5 and using (2.61), we have

$$\langle \nabla_{j_k} f(\mathbf{x}^k), \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{k+1} \rangle \le f(\mathbf{x}^k) - f(\mathbf{x}^{k+1}) + \frac{L}{2}\|\mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{k+1}\|_2^2 \,. \tag{2.81}$$

Letting $\mathbf{x} = \mathbf{x}^*$ and using the smoothness of f, we have

$$\begin{aligned}&\langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k} \rangle + g_{j_k}(\mathbf{x}^k) - g_{j_k}(\mathbf{x}^*) \\ &\le \langle \mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k), \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{k+1} \rangle + f(\mathbf{x}^k) + g(\mathbf{x}^k) - [f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})] \\ &+ \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^{k}\|_2^2 - \|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2 - \|\mathbf{x}_{j_k}^{k+1} - \mathbf{x}_{j_k}^{k}\|_2^2) + \frac{L}{2}\|\mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{k+1}\|_2^2 \\ &\le \frac{1}{2(\eta - L)}\|\mathbf{v}_{j_k}^{i_k} - \nabla_{j_k} f(\mathbf{x}^k)\|_2^2 + f(\mathbf{x}^k) + g(\mathbf{x}^k) - [f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})] \\ &+ \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^{k}\|_2^2 - \|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2) \,.\end{aligned} \tag{2.82}$$

Taking expectation over i_k, j_k on both sides and using Lemma 4, we have

$$\begin{aligned}&\mathbb{E}[\langle \mathbf{v}_{j_k}^{i_k}, \mathbf{x}_{j_k}^{k} - \mathbf{x}_{j_k}^{*} \rangle + g_{j_k}(\mathbf{x}^k) - g_{j_k}(\mathbf{x}^*)] \\ &\le \frac{L}{J(\eta - L)}[h(\mathbf{x}^k, \mathbf{x}^*) + h(\tilde{\mathbf{x}}, \mathbf{x}^*)] + f(\mathbf{x}^k) + g(\mathbf{x}^k) - \mathbb{E}[f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})] \\ &+ \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^{k}\|_2^2 - \mathbb{E}\|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2) \,.\end{aligned} \tag{2.83}$$

The left hand side can be rewritten as

$$\begin{aligned}&\mathbb{E}[\langle \mathbf{v}^{i_k}_{j_k}, \mathbf{x}^k_{j_k} - \mathbf{x}^*_{j_k}\rangle + g_{j_k}(\mathbf{x}^k) - g_{j_k}(\mathbf{x}^*)] = \frac{1}{J}[\mathbb{E}_{i_k}\langle \mathbf{v}^{i_k}, \mathbf{x}^k - \mathbf{x}^*\rangle + g(\mathbf{x}^k) - g(\mathbf{x}^*)]\\ &= \frac{1}{J}[\langle \nabla f(\mathbf{x}^k), \mathbf{x}^k - \mathbf{x}^*\rangle + g(\mathbf{x}^k) - g(\mathbf{x}^*)] = \frac{1}{J}h(\mathbf{x}^k, \mathbf{x}^*) \ . \end{aligned} \tag{2.84}$$

Plugging into (2.83) gives

$$\begin{aligned}\frac{1}{J}[h(\mathbf{x}^k, \mathbf{x}^*)] &\le \frac{L}{J(\eta - L)}[h(\mathbf{x}^k, \mathbf{x}^*) + h(\tilde{\mathbf{x}}, \mathbf{x}^*)] + f(\mathbf{x}^k) + g(\mathbf{x}^k) - \mathbb{E}[f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})]\\ &\quad + \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^k\|_2^2 - \mathbb{E}\|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2)\\ &\le \frac{L}{J(\eta - L)}[h(\mathbf{x}^k, \mathbf{x}^*) + h(\tilde{\mathbf{x}}, \mathbf{x}^*)] + f(\mathbf{x}^k) + g(\mathbf{x}^k) - \mathbb{E}[f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})]\\ &\quad + \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^k\|_2^2 - \mathbb{E}\|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2) \ ,\end{aligned} \tag{2.85}$$

Rearranging the terms yields

$$\begin{aligned}\frac{\eta - 2L}{J(\eta - L)}h(\mathbf{x}^k, \mathbf{x}^*) &\le \frac{L}{J(\eta - L)}[h(\tilde{\mathbf{x}}, \mathbf{x}^*)] + f(\mathbf{x}^k) + g(\mathbf{x}^k) - \mathbb{E}[f(\mathbf{x}^{k+1}) + g(\mathbf{x}^{k+1})]\\ &\quad + \frac{\eta}{2}(\|\mathbf{x}^* - \mathbf{x}^k\|_2^2 - \mathbb{E}\|\mathbf{x}^* - \mathbf{x}^{k+1}\|_2^2) \ .\end{aligned} \tag{2.86}$$

At time $t+1$, we have $\mathbf{x}_0 = \tilde{\mathbf{x}} = \mathbf{x}^t$. Summing over $k = 0, \cdots, m-1$ and taking expectation with respect to the history of random variable ξ, we have

$$\begin{aligned}\frac{\eta - 2L}{J(\eta - L)}\sum_{k=0}^{m-1}\mathbb{E}_\xi h(\mathbf{x}_k, \mathbf{x}^*) &\le \frac{Lm}{J(\eta - L)}\mathbb{E}_\xi h(\tilde{\mathbf{x}}, \mathbf{x}^*) + \mathbb{E}_\xi[f(\mathbf{x}_0) + g(\mathbf{x}_0)] - \mathbb{E}_\xi[f(\mathbf{x}_m) + g(\mathbf{x}_m)]\\ &\quad + \frac{\eta}{2}(\mathbb{E}_\xi\|\mathbf{x}^* - \mathbf{x}_0\|_2^2 - \mathbb{E}_\xi\|\mathbf{x}^* - \mathbf{x}_m\|_2^2)\\ &\le \frac{Lm}{J(\eta - L)}\mathbb{E}_\xi h(\tilde{\mathbf{x}}, \mathbf{x}^*) + \mathbb{E}_\xi h(\mathbf{x}_0, \mathbf{x}^*)\\ &\quad + \frac{\eta}{2}(\mathbb{E}_\xi\|\mathbf{x}^* - \mathbf{x}_0\|_2^2 - \mathbb{E}_\xi\|\mathbf{x}^* - \mathbf{x}_m\|_2^2) \ ,\end{aligned}$$

where the last inequality uses

$$\begin{aligned}f(\mathbf{x}_0) + g(\mathbf{x}_0) - [f(\mathbf{x}_m) + g(\mathbf{x}_m)] &\le f(\mathbf{x}_0) + g(\mathbf{x}_0) - [f(\mathbf{x}^*) + g(\mathbf{x}^*)]\\ &\le \langle \nabla f(\mathbf{x}_0), \mathbf{x}_0 - \mathbf{x}^*\rangle + g(\mathbf{x}_0) - g(\mathbf{x}^*)\\ &= h(\mathbf{x}_0, \mathbf{x}^*) \ .\end{aligned} \tag{2.87}$$

Rearranging the terms gives

$$\frac{\eta-2L}{J(\eta-L)}\sum_{k=1}^{m-1}\mathbb{E}_\xi h(\mathbf{x}_k,\mathbf{x}^*)+\frac{\eta}{2}\mathbb{E}_\xi\|\mathbf{x}^*-\mathbf{x}_m\|_2^2$$
$$\le\frac{Lm}{J(\eta-L)}\mathbb{E}_\xi h(\tilde{\mathbf{x}},\mathbf{x}^*)+(1-\frac{\eta-2L}{J(\eta-L)})\mathbb{E}_\xi h(\mathbf{x}_0,\mathbf{x}^*)+\frac{\eta}{2}\mathbb{E}_\xi\|\mathbf{x}^*-\mathbf{x}_0\|_2^2\,. \tag{2.88}$$

According to Lemma 3 and assuming $\eta > 2L$, we have

$$\frac{\eta-2L}{J(\eta-L)}\sum_{k=1}^{m}\mathbb{E}_\xi h(\mathbf{x}_k,\mathbf{x}^*)\le\frac{Lm}{J(\eta-L)}\mathbb{E}_\xi h(\tilde{\mathbf{x}},\mathbf{x}^*)+(1-\frac{\eta-2L}{J(\eta-L)})\mathbb{E}_\xi h(\mathbf{x}_0,\mathbf{x}^*)$$
$$+\frac{\eta}{2}\mathbb{E}_\xi\|\mathbf{x}^*-\mathbf{x}_0\|_2^2\,. \tag{2.89}$$

Pick x^{t+1} so that $h(\mathbf{x}^{t+1})\le h(\mathbf{x}_k), 1\le k\le m-1$, we have

$$\frac{\eta-2L}{J(\eta-L)}m\mathbb{E}_\xi h(\mathbf{x}^{t+1},\mathbf{x}^*)\le[\frac{Lm}{J(\eta-L)}+1-\frac{\eta-2L}{J(\eta-L)}]\mathbb{E}_\xi h(\mathbf{x}^t,\mathbf{x}^*)+\frac{\eta}{2}\mathbb{E}_\xi\|\mathbf{x}^*-\mathbf{x}^t\|_2^2\,, \tag{2.90}$$

where ther right hand side uses $\mathbf{x}^t=\mathbf{x}_0=\tilde{\mathbf{x}}$. Using (2.64), we have

$$\frac{\eta-2L}{J(\eta-L)}m\mathbb{E}_\xi h(\mathbf{x}^{t+1},\mathbf{x}^*)\le[\frac{Lm}{J(\eta-L)}+1-\frac{\eta-2L}{J(\eta-L)}+\frac{\eta}{\gamma}]\mathbb{E}_\xi h(\mathbf{x}^t,\mathbf{x}^*)\,. \tag{2.91}$$

Dividing both sides by $\frac{\eta-2L}{J(\eta-L)}m$, we have

$$\mathbb{E}_\xi h(\mathbf{x}^{t+1},\mathbf{x}^*)\le\rho\mathbb{E}_\xi h(\mathbf{x}^t,\mathbf{x}^*)\,, \tag{2.92}$$

where

$$\rho=\frac{L}{\eta-2L}+\frac{(\eta-L)J}{(\eta-2L)m}-\frac{1}{m}+\frac{\eta(\eta-L)J}{(\eta-2L)m\gamma}<1\,, \tag{2.93}$$

which completes the proof. ■

Part II

Equality-constrained Optimization

Chapter 3

Alternating Direction Method of Multipliers

3.1 Introduction

In this chapter, we consider optimization problems of the following form:

$$\min_{\mathbf{x}\in\mathcal{X},\mathbf{z}\in\mathcal{Z}} f(\mathbf{x})+g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}\ , \tag{3.1}$$

where $\mathbf{A}\in\mathbb{R}^{m\times n_1}, \mathbf{B}\in\mathbb{R}^{m\times n_2}, \mathbf{c}\in\mathbb{R}^m$, $\mathbf{x}\in\mathcal{X}\in\mathbb{R}^{n_1\times 1}, \mathbf{z}\in\mathcal{Z}\in\mathbb{R}^{n_2\times 1}$ and $\mathcal{X}$ and $\mathcal{Z}$ are convex sets. The linear equality constraint introduces splitting variables and thus splits functions and feasible sets into simpler constraint sets $\mathbf{x}\in\mathcal{X}$ and $\mathbf{z}\in\mathcal{Z}$. (6.2) can easily accommodate linear inequality constraints by introducing a slack variable. In the sequel, we drop the convex sets $\mathcal{X}$ and $\mathcal{Z}$ for ease of exposition, noting that one can consider g and other additive functions to be the indicators of suitable convex feasible sets. f and g can be non-smooth, including piecewise linear and indicator functions. In the context of machine learning, f is usually a loss function such as ℓ_1, ℓ_2, hinge and logistic loss, while g is a regularizer, e.g., ℓ_1, ℓ_2, nuclear norm, mixed-norm and total variation.

(3.1) can be solved by the well known alternating direction method of multipliers (ADMM or ADM) [19]. In each iteration, ADMM updates splitting variables separately and alternatively by solving the augmented Lagrangian of (3.1), which is defined as follows:

$$L_\rho(\mathbf{x},\mathbf{z},\mathbf{y})=f(\mathbf{x})+g(\mathbf{z})+\langle\mathbf{y},\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}-\mathbf{c}\rangle+\frac{\rho}{2}\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}-\mathbf{c}\|_2^2, \tag{3.2}$$

where $\mathbf{y} \in \mathbb{R}^m$ is dual variable, $\rho > 0$ is penalty parameter, and the quadratic penalty term is to penalize the violation of the equality constraint. ADMM consists of the following three updates:

$$\mathbf{x}_{t+1} = \operatorname{argmin}_{\mathbf{x}} \ f(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c} \rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \ , \tag{3.3}$$

$$\mathbf{z}_{t+1} = \operatorname{argmin}_{\mathbf{z}} \ g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c}\|_2^2 \ , \tag{3.4}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \ . \tag{3.5}$$

First introduced in [67], ADM has since been extensively explored in recent years due to its ease of applicability and empirical performance in a wide variety of problems, including composite objectives [19, 55, 116]. It has been shown as a special case of Douglas-Rachford splitting method [37, 49, 55], which in turn is a special case of the proximal point method [166]. Recent literature has illustrated the empirical efficiency of ADM in a broad spectrum of applications ranging from image processing [149, 58, 1, 28] to applied statistics and machine learning [172, 1, 219, 220, 214, 116, 8, 139, 133]. ADM has been shown to outperform state-of-the-art methods for sparse problems, including LASSO [189, 74, 1, 19], total variation [69], sparse inverse covariance selection [45, 6, 60, 136, 172, 219], and sparse and low rank approximations [220, 116, 24]. ADM have also been used to solve linear programs (LPs) [54], LP decoding [8] and MAP inference problems in graphical models [133, 139, 64]. In addition, an advantage of ADM is that it can handle linear equality constraint of the form $\{\mathbf{x}, \mathbf{z} | \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}\}$, which makes distributed optimization by variable splitting in a batch setting straightforward [15, 142, 19, 154]. For further understanding of ADM, we refer the readers to the comprehensive review by [19] and references therein.

Although the proof of global convergence of ADM can be found in [66, 55, 19], the literature does not have the convergence rate for ADM. We introduce proof techniques for the rate of convergence of ADM in the batch setting, which establish a $O(1/T)$ convergence rate for the objective, the optimality conditions (constraints) and ADM based on variational inequalities [56]. The $O(1/T)$ convergence rate for ADM is in line with gradient methods for composite objective [145, 146, 53][1] . Our proof requires rather weak assumptions compared to the Lipschitz continuous gradient required in general in gradient methods [145, 146, 53]. During/after the publication of our preliminary version [200], the convergence rate for ADM was also shown

[1] The gradient methods can be accelerated to achieve the $O(1/T^2)$ convergence rate [145, 146].

in [80, 79, 86, 48, 17, 70]. For strongly convex functions, the dual objective of an accelerated version of ADMM can converge at a rate of $O(1/T^2)$ [70]. For strongly convex functions, ADMM can achieve a linear convergence rate [48]. Our proof is different and self-contained. In particular, the other approaches do not prove the convergence rate for the objective, which is fundamentally important to regret analysis in the online setting.

3.2 Analysis for Batch Alternating Direction Method

We are interested in the rate of convergence of ADM in terms of iteration complexity, i.e., the number of iterations needed to obtain an ϵ-optimal solution. Most first-order methods require functions to be smooth or having Lipschitz continuous gradient to establish the convergence rate [145, 146, 53]. The assumptions in establishing convergence rate of ADM are relatively simple [19], and are stated below for the sake of completeness:

Assumption 4
(a) $f : \mathbb{R}^{n_1} \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^{n_2} \to \mathbb{R} \cup \{+\infty\}$ are closed, proper and convex.
(b) A KKT point $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ of the Lagrangian (3.2) of the problem (3.1) exists.

We first analyze the convergence rate for the objective and optimality conditions (constraints) separately, which play an important role for the regret analysis in the online setting. Then, the rate of convergence is established under a joint analysis of the objective and constraints using a variational inequality [56].

3.2.1 Convergence Rate for the Objective

The updates of $\mathbf{x}, \mathbf{z}$ implicitly generate the (sub)gradients of $f(\mathbf{x}_{t+1})$ and $g(\mathbf{z}_{t+1})$, as given in the following lemma.

Lemma 6 *Let $\partial f(\mathbf{x}_{t+1})$ be the subgradient of $f(\mathbf{x})$ at $\mathbf{x}_{t+1}$, we have*

$$-\mathbf{A}^T(\mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c})) \in \partial f(\mathbf{x}_{t+1}) \, , \tag{3.6}$$

$$-\mathbf{A}^T(\mathbf{y}_{t+1} + \rho(\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1})) \in \partial f(\mathbf{x}_{t+1}) \tag{3.7}$$

Let $\partial g(\mathbf{z}_{t+1})$ be the subgradient of $g(\mathbf{z})$ at $\mathbf{z}_{t+1}$, we have

$$-\mathbf{B}^T\mathbf{y}_{t+1} \in \partial g(\mathbf{z}_{t+1}) \, . \tag{3.8}$$

Proof: Since $\mathbf{x}_{t+1}$ minimizes (3.3), we have

$$0 \in \partial f(\mathbf{x}_{t+1}) + \mathbf{A}^T\mathbf{y}_t + \rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t - \mathbf{c}) \ .$$

Rearranging the terms gives (3.6). Using (3.5) yields (3.7).

Similarly, $\mathbf{z}_{t+1}$ minimizes (3.4), then

$$\partial g(\mathbf{z}_{t+1}) + \mathbf{B}^T\mathbf{y}_t + \rho\mathbf{B}^T(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \in 0 \ .$$

Rearranging the terms and using (3.5) yield (3.8). ■

The following lemma shows the inaccuracy of the objective with respect to the optimum at $(t+1)$ is bounded by step differences of $\mathbf{y}$ and $\mathbf{z}$.

Lemma 7 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by ADM. Then for any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, we have*

$$\begin{aligned}
&f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \\
&\leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \ .
\end{aligned} \tag{3.9}$$

Proof: Since $f(\mathbf{x})$ is a convex function and its subgradient is given in (3.7),

$$\begin{aligned}
f(\mathbf{x}_{t+1}) - f(\mathbf{x}^*) &\leq -\langle \mathbf{A}^T(\mathbf{y}_{t+1} + \rho(\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1})), \mathbf{x}_{t+1} - \mathbf{x}^* \rangle \\
&= -\langle \mathbf{y}_{t+1} + \rho(\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1}), \mathbf{A}\mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}^* \rangle \\
&= -\langle \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} - \mathbf{c} + \mathbf{B}\mathbf{z}^* \rangle + \rho\langle \mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t, \mathbf{A}\mathbf{x}_{t+1} - \mathbf{c} + \mathbf{B}\mathbf{z}^* \rangle \\
&= -\langle \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} - \mathbf{c} + \mathbf{B}\mathbf{z}^* \rangle + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2 \\
&\quad + \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 - \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2) \ .
\end{aligned} \tag{3.10}$$

where the last equality uses

$$\langle \mathbf{u}_1 - \mathbf{u}_2, \mathbf{u}_3 + \mathbf{u}_4 \rangle = \frac{1}{2}(\|\mathbf{u}_4 - \mathbf{u}_2\|_2^2 - \|\mathbf{u}_4 - \mathbf{u}_1\|_2^2 + \|\mathbf{u}_3 + \mathbf{u}_1\|_2^2 - \|\mathbf{u}_3 + \mathbf{u}_2\|_2^2). \tag{3.11}$$

Similarly, for convex function $g(\mathbf{z})$, using its subgradient in (3.8), we have

$$g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) \leq -\langle \mathbf{B}^T\mathbf{y}_{t+1}, \mathbf{z}_{t+1} - \mathbf{z}^* \rangle = -\langle \mathbf{y}_{t+1}, \mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}^* \rangle \ . \tag{3.12}$$

Adding (3.10) and (3.12) together yields

$$\begin{aligned}&f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))\\&\le -\langle \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2\\&- \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) .\end{aligned} \tag{3.13}$$

Recalling (3.5), the first two terms in (3.13) can be rewritten as

$$\begin{aligned}&- \langle \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2\\&= \frac{1}{2\rho}(2\langle \mathbf{y}_{t+1}, \mathbf{y}_t - \mathbf{y}_{t+1}\rangle + \|\mathbf{y}_t - \mathbf{y}_{t+1}\|_2^2)\\&= \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) .\end{aligned} \tag{3.14}$$

Plugging back into (3.13) yields the result. ∎

As observed in several experiments [19], the objective is not monotonically non-increasing. The following theorem shows the objective of ADM has the $O(1/T)$ convergence rate in an ergodic sense.

Theorem 6 *Let the sequences* $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ *be generated by ADM and* $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{x}_t, \bar{\mathbf{z}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{z}_t$. *For any* $\mathbf{x}^*, \mathbf{z}^*$ *satisfying* $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, *for any* T, *we have*

$$f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \le \frac{\frac{1}{\rho}\|\mathbf{y}_0\|_2^2 + \rho\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_0\|_2^2}{2T} . \tag{3.15}$$

Proof: In (3.9), ignoring $-\frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$ and summing over t from 0 to $T-1$, we have the following telescoping sum

$$\begin{aligned}&\sum_{t=0}^{T-1} [f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))]\\&\le \frac{1}{2\rho}(\|\mathbf{y}_0\|_2^2 - \|\mathbf{y}_T\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_0\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_T\|_2^2) .\end{aligned}$$

Since both f and g are convex, dividing by T, applying Jensen's inequality and letting the assumptions hold complete the proof. ∎

Although (3.15) shows that the objective value converges to the optimal value, $\{\mathbf{x}_{t+1}, \mathbf{z}_{t+1}\}$ may not be feasible and the equality constraint may not necessarily be satisfied.

3.2.2 Convergence Rate for the Optimality Conditions (Constraints)

Assume that $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ satisfies the KKT conditions of the Lagrangian (3.2), i.e.,

$$-\mathbf{A}^T\mathbf{y}^* \in \partial f(\mathbf{x}^*) \ , \tag{3.16}$$

$$-\mathbf{B}^T\mathbf{y}^* \in \partial g(\mathbf{z}^*) \ , \tag{3.17}$$

$$\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* - \mathbf{c} = 0 \ . \tag{3.18}$$

According to (3.7), condition (3.16) holds if $\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t = 0$. According to (3.8), condition (3.17) holds for every iterate. Therefore, the KKT conditions (3.16)-(3.18) hold if the following optimality conditions are satisfied:

$$\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t = 0 \ , \tag{3.19}$$

$$\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} = 0 \ , \tag{3.20}$$

The LHS of (3.19) is called *primal residual* and the LHS of (3.20) is called equality constraint violation or *dual residual* [19] when considering (3.5).

Define a residual function of optimality conditions as

$$R(s,t) = \|\mathbf{A}\mathbf{x}_s + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \|\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{s-1}\|_2^2 \ , \tag{3.21}$$

where $s \in \{t, t+1\}$. In particular, the residual after the $\mathbf{z}$ update (3.4) at iteration $(t+1)$ is

$$R(t+1,t+1) = \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 + \|\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t\|_2^2 \ . \tag{3.22}$$

and the residual after the $\mathbf{x}$-update (3.3) at $(t+1)$ is

$$R(t+1,t) = \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \ . \tag{3.23}$$

Therefore, the convergence of $R(t+1,t+1)$ implies the convergence of the optimality conditions.

The following two lemmas show the residuals of optimality conditions (constraints) are monotonically non-increasing.

Lemma 8 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by ADM. Then*

$$R(t+1,t) \leq R(t,t) \tag{3.24}$$

Proof: Since $f(\mathbf{x})$ is a convex function and its subgradient is given in (3.6), for any $\mathbf{x}$, we have

$$\begin{aligned} f(\mathbf{x}_{t+1}) - f(\mathbf{x}) &\le -\langle \mathbf{A}^T(\mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c})), \mathbf{x}_{t+1} - \mathbf{x}\rangle \\ &= \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}\rangle + \rho\langle \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}, \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}\rangle \ . \end{aligned} \tag{3.25}$$

Letting $\mathbf{x} = \mathbf{x}_t$, we have

$$\begin{aligned} &f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t) \le \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t+1}\rangle + \rho\langle \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}, \mathbf{A}\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t+1}\rangle \\ &= \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t+1}\rangle + \frac{\rho}{2}(\|\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 - \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 - \|\mathbf{A}\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t+1}\|_2^2) \ . \end{aligned} \tag{3.26}$$

where the last equality uses

$$\langle \mathbf{u}_1 - \mathbf{u}_2, \mathbf{u}_3 - \mathbf{u}_1\rangle = \frac{1}{2}(\|\mathbf{u}_2 - \mathbf{u}_3\|_2^2 - \|\mathbf{u}_1 - \mathbf{u}_2\|_2^2 - \|\mathbf{u}_1 - \mathbf{u}_3\|_2^2) \ . \tag{3.27}$$

Using the subgradient of f given in (3.7) at $\mathbf{x}_t$, for any $\mathbf{x}$,

$$f(\mathbf{x}_t) - f(\mathbf{x}) \le -\langle \mathbf{A}^T(\mathbf{y}_t + \rho(\mathbf{B}\mathbf{z}_{t-1} - \mathbf{B}\mathbf{z}_t)), \mathbf{x}_t - \mathbf{x}\rangle \ . \tag{3.28}$$

Letting $\mathbf{x} = \mathbf{x}_{t+1}$, we have

$$\begin{aligned} f(\mathbf{x}_t) - f(\mathbf{x}_{t+1}) &\le -\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t+1}\rangle + \rho\langle \mathbf{B}\mathbf{z}_{t-1} - \mathbf{B}\mathbf{z}_t, \mathbf{A}\mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}_t\rangle \\ &\le \langle \mathbf{A}\mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}_t, \mathbf{y}_t\rangle + \frac{\rho}{2}(\|\mathbf{A}\mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}_t\|_2^2 + \|\mathbf{B}\mathbf{z}_{t-1} - \mathbf{B}\mathbf{z}_t\|_2^2) \ . \end{aligned} \tag{3.29}$$

Adding (3.26) and (3.29) together and rearranging the terms complete the proof. ■

Lemma 9 *Let the sequences* $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ *be generated by ADM. Then*

$$R(t+1, t+1) \le R(t+1, t) \tag{3.30}$$

Proof: Recalling the subgradient of convex function $g(\mathbf{z})$ given in (3.8), we have

$$g(\mathbf{z}_{t+1}) - g(\mathbf{z}_t) \le \langle -\mathbf{B}^T\mathbf{y}_{t+1}, \mathbf{z}_{t+1} - \mathbf{z}_t\rangle \ , \tag{3.31}$$

$$g(\mathbf{z}_t) - g(\mathbf{z}_{t+1}) \le \langle -\mathbf{B}^T\mathbf{y}_t, \mathbf{z}_t - \mathbf{z}_{t+1}\rangle \ . \tag{3.32}$$

Adding (3.31) and (3.32) together yields

$$0 \leq \langle \mathbf{B}^T(\mathbf{y}_{t+1} - \mathbf{y}_t), \mathbf{z}_t - \mathbf{z}_{t+1} \rangle = \langle \mathbf{y}_{t+1} - \mathbf{y}_t, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle \ . \tag{3.33}$$

According to (3.5), the right-hand side can be rewritten as

$$\begin{aligned} &\langle \mathbf{y}_{t+1} - \mathbf{y}_t, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle \\ &= \rho \langle \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}, (\mathbf{B}\mathbf{z}_t - \mathbf{c}) - (\mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \rangle \\ &= \frac{\rho}{2} \big(\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 - \|\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \big) \ . \end{aligned} \tag{3.34}$$

Plugging into (3.33) and rearranging the terms complete the proof. ■

The above two lemmas together shows that

$$R(t+1, t+1) \leq R(t+1, t) \leq R(t, t) \leq R(t, t-1) \ , \tag{3.35}$$

meaning $R(s, t)$ is monotonically non-increasing. The following lemma shows $R(t+1, t)$ is bounded by step differences of a telescoping series of $\mathbf{y}$ and $\mathbf{z}$.

Lemma 10 *Let the sequences* $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ *be generated by ADM and* $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ *satisfy the KKT conditions (3.16)-(3.18), then*

$$R(t+1, t) \leq \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2 + \frac{1}{\rho^2} \left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2 \right) \ . \tag{3.36}$$

Proof: Assume $\{\mathbf{x}^*, \mathbf{y}^*\}$ satisfies (3.16). Since f is convex, then

$$f(\mathbf{x}^*) - f(\mathbf{x}_{t+1}) \leq -\langle \mathbf{A}^T \mathbf{y}^*, \mathbf{x}^* - \mathbf{x}_{t+1} \rangle = -\langle \mathbf{y}^*, \mathbf{A}\mathbf{x}^* - \mathbf{A}\mathbf{x}_{t+1} \rangle \ . \tag{3.37}$$

Similarly, for convex function g and $\{\mathbf{z}^*, \mathbf{y}^*\}$ satisfies (3.17), we have

$$g(\mathbf{z}^*) - g(\mathbf{z}_{t+1}) \leq -\langle \mathbf{B}^T \mathbf{y}^*, \mathbf{z}^* - \mathbf{z}_{t+1} \rangle = -\langle \mathbf{y}^*, \mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1} \rangle \ . \tag{3.38}$$

Adding them together and using the fact that $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, we have

$$f(\mathbf{x}^*) + g(\mathbf{z}^*) - (f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1})) \leq \langle \mathbf{y}^*, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} \rangle \ . \tag{3.39}$$

Adding (3.13) and (3.39) together yields

$$0 \leq \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2 - \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2)$$

$$+ \langle \mathbf{y}^* - \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} \rangle \ . \tag{3.40}$$

The last term can be rewritten as

$$\langle \mathbf{y}^* - \mathbf{y}_{t+1}, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} \rangle = \frac{1}{\rho}\langle \mathbf{y}^* - \mathbf{y}_{t+1}, \mathbf{y}_{t+1} - \mathbf{y}_t \rangle$$
$$= -\frac{1}{2\rho}\left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2 - \|\mathbf{y}_{t+1} - \mathbf{y}_t\|_2^2\right) \ . \tag{3.41}$$

Substituting it into (3.40) and rearranging the terms gives

$$\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2 + \frac{1}{\rho^2}\left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2\right)$$
$$\geq \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 - \frac{1}{\rho^2}\|\mathbf{y}_{t+1} - \mathbf{y}_t\|_2^2$$
$$= \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \ , \tag{3.42}$$

which completes the proof. ■

Now, we are ready to show that the optimality conditions have a $O(1/T)$ convergence rate.

Theorem 7 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by ADM. For any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, for any T, we have*

$$R(T,T) \leq R(T,T-1) \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + D_{\mathbf{y}}^2/\rho^2}{T} \ , \tag{3.43}$$

where $R(T,T) = \|\mathbf{A}\mathbf{x}_T + \mathbf{B}\mathbf{z}_T - \mathbf{c}\|_2^2 + \|\mathbf{B}\mathbf{z}_T - \mathbf{B}\mathbf{z}_{T-1}\|_2^2$.

Proof: Since $\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$ is monotonically non-increasing, we have

$$TR(T,T-1) = T\|\mathbf{A}\mathbf{x}_T + \mathbf{B}\mathbf{z}_{T-1} - \mathbf{c}\|_2^2 \leq \sum_{t=0}^{T-1} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$$
$$\leq \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_0\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_T\|_2^2 + \frac{1}{\rho^2}(\|\mathbf{y}^* - \mathbf{y}_0\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_T\|_2^2)$$
$$\leq \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_0\|_2^2 + \frac{1}{\rho^2}\|\mathbf{y}^* - \mathbf{y}_0\|_2^2 \ . \tag{3.44}$$

Divide both sides by T. Letting Assumption 4 hold and using Lemma 9 yield (3.43). ■

Results similar to Lemma 9 and 10 have appeared in [19], but Lemma 8 is new. The monotonicity and $O(1/T)$ convergence rate for optimality conditions have also been shown in [79], but our proof is different and self-contained.

3.2.3 Rate of Convergence of ADM based on Variational Inequality

We now prove the $O(1/T)$ convergence rate for ADM using a variational inequality (VI) based on the Lagrangian given in (3.2). In this section, we need the following assumption [14, 13]:

Assumption 5 $\mathbf{y}$ *is bounded in* $\mathbb{R}^m$ *and* $\|\mathbf{y}\|_2 \leq D$*, i.e.,* $\mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^m$ *and* $\mathcal{Y}$ *is a bounded set.*

Let $\Omega = \mathcal{X} \times \mathcal{Z} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Z}$ are defined in (6.2). Any $\mathbf{w}^* = (\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*) \in \Omega$ solves the original problem in (3.1) optimally if it satisfies the following variational inequality [56, 143, 80]:

$$\forall \mathbf{w} \in \Omega \,, \quad h(\mathbf{w}) - h(\mathbf{w}^*) + \langle \mathbf{w} - \mathbf{w}^*, F(\mathbf{w}^*) \rangle \geq 0 \,, \tag{3.45}$$

where $h(\mathbf{w}) = f(\mathbf{x}) + g(\mathbf{z})$ and

$$F(\mathbf{w}) = \begin{bmatrix} \mathbf{A}^T\mathbf{y} \\ \mathbf{B}^T\mathbf{y} \\ -(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{A}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{B}^T \\ -\mathbf{A} & -\mathbf{B} & \mathbf{0} \end{bmatrix} \mathbf{w} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{c} \end{bmatrix} = \mathbf{M}\mathbf{w} + \mathbf{q}$$

is the gradient of the last term of the Lagrangian. $\mathbf{M}$ is an anti-symmetric matrix and $\mathbf{w}^T\mathbf{M}\mathbf{w} = 0$. Then, $\tilde{\mathbf{w}} = (\tilde{\mathbf{x}}, \tilde{\mathbf{z}}, \tilde{\mathbf{y}})$ approximately solves the problem with accuracy ϵ if it satisfies

$$\forall \mathbf{w} \in \Omega \,, \quad h(\tilde{\mathbf{w}}) - h(\mathbf{w}) + \langle \tilde{\mathbf{w}} - \mathbf{w}, F(\tilde{\mathbf{w}}) \rangle \leq \epsilon \,. \tag{3.46}$$

We show that after T iterations, the average $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{w}_t$, where $\mathbf{w}_t = (\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t)$ are from (3.3)-(3.5), satisfies the above inequality with $\epsilon = O(1/T)$.

Theorem 8 *Let* $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{w}_t$*, where* $\mathbf{w}_t = (\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t)$ *from (3.3)-(3.5). Let Assumption 4 and 5 hold, then*

$$\forall \mathbf{w} \in \Omega, \quad h(\bar{\mathbf{w}}_T) - h(\mathbf{w}) + \langle \bar{\mathbf{w}}_T - \mathbf{w}, F(\bar{\mathbf{w}}_T) \rangle \leq \frac{L}{T} \,.$$

where $L = \frac{\rho}{2}\|\mathbf{A}\mathbf{x} - \mathbf{c}\|_2^2 + \frac{1}{2\rho}\|\mathbf{y}\|^2$.

Proof: Considering $f(\mathbf{x})$ is a convex function and its subgradient is given in (3.7), $\forall \mathbf{x} \in \mathcal{X}$,

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}) \leq -\langle \mathbf{A}^T(\mathbf{y}_{t+1} + \rho(\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1})), \mathbf{x}_{t+1} - \mathbf{x} \rangle \,.$$

Rearranging the terms gives

$$f(\mathbf{x}_{t+1}) - f(\mathbf{x}) + \langle \mathbf{x}_{t+1} - \mathbf{x}, \mathbf{A}^T\mathbf{y}_{t+1} \rangle \leq \rho \langle \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle \,. \tag{3.47}$$

Using the subgradient of g given in (3.8), we have $\forall \mathbf{z} \in \mathcal{Z}$

$$g(\mathbf{z}_{t+1}) - g(\mathbf{z}) + \langle \mathbf{z}_{t+1} - \mathbf{z}, \mathbf{B}^T \mathbf{y}_{t+1} \rangle \leq 0 \ . \tag{3.48}$$

Adding (3.47) and (3.48) and denoting $h(\mathbf{w}) = f(\mathbf{x}) + g(\mathbf{z})$, we have $\forall \mathbf{w} \in \Omega$

$$\begin{aligned} & h(\mathbf{w}_{t+1}) - h(\mathbf{w}) + \langle \mathbf{w}_{t+1} - \mathbf{w}, F(\mathbf{w}_{t+1}) \rangle \qquad (3.49) \\ & \leq \ \rho \langle \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle + \frac{1}{\rho} \langle \mathbf{y}_{t+1} - \mathbf{y}, -(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \rangle \\ & = \ \rho \langle \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle + \frac{1}{\rho} \langle \mathbf{y} - \mathbf{y}_{t+1}, \mathbf{y}_{t+1} - \mathbf{y}_t \rangle \ . \end{aligned}$$

The first term can be rewritten as

$$\begin{aligned} & 2\langle \mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle \qquad (3.50) \\ & = 2\langle \mathbf{A}\mathbf{x} - \mathbf{c} - (\mathbf{A}\mathbf{x}_{t+1} - \mathbf{c}), \mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1} \rangle \\ & = \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|^2 - \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|^2 + \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|^2 - \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|^2 \ , \end{aligned}$$

where the last equality uses (8.25). The second term in (3.49) is equivalent to

$$2\langle \mathbf{y} - \mathbf{y}_{t+1}, \mathbf{y}_{t+1} - \mathbf{y}_t \rangle = \|\mathbf{y} - \mathbf{y}_t\|^2 - \|\mathbf{y} - \mathbf{y}_{t+1}\|^2 - \|\mathbf{y}_t - \mathbf{y}_{t+1}\|^2 \ , \tag{3.51}$$

which uses (8.24). Substituting (3.50) and (3.51) into (3.49) and using (3.5), we have

$$\begin{aligned} & h(\mathbf{w}_{t+1}) - h(\mathbf{w}) + \langle \mathbf{w}_{t+1} - \mathbf{w}, F(\mathbf{w}_{t+1}) \rangle \\ & \leq \frac{\rho}{2} (\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|^2 - \|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|^2) + \frac{1}{2\rho} (\|\mathbf{y} - \mathbf{y}_t\|^2 - \|\mathbf{y} - \mathbf{y}_{t+1}\|^2) \ . \end{aligned} \tag{3.52}$$

Summing over t from 0 to $T-1$, we have the following telescoping sum

$$\sum_{t=1}^{T} [h(\mathbf{w}_t) - h(\mathbf{w}) + \langle \mathbf{w}_t - \mathbf{w}, F(\mathbf{w}_t) \rangle] \leq L \ , \tag{3.53}$$

where the constant $L = \frac{\rho}{2}\|\mathbf{A}\mathbf{x} - \mathbf{c}\|_2^2 + \frac{1}{2\rho}\|\mathbf{y}\|^2$. Recall that $h(\tilde{w})$ is a convex function of $\tilde{w}$. Further, from the definition of $F(\tilde{\mathbf{w}})$, we have

$$\langle \tilde{\mathbf{w}} - \mathbf{w}, F(\tilde{\mathbf{w}}) \rangle = \langle \tilde{\mathbf{w}} - \mathbf{w}, \mathbf{M}\tilde{\mathbf{w}} + \mathbf{q} \rangle = -\langle \mathbf{w}, \mathbf{M}\tilde{\mathbf{w}} \rangle + \langle \tilde{\mathbf{w}} - \mathbf{w}, \mathbf{q} \rangle \ , \tag{3.54}$$

which is a linear function of $\tilde{\mathbf{w}}$. Dividing both sides of (3.53) by T, recalling that $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^{T}\mathbf{w}_t$, and using Jensen's inequality, we have

$$\begin{aligned}
&h(\bar{\mathbf{w}}_T) - h(\mathbf{w}) + \langle \bar{\mathbf{w}}_T - \mathbf{w}, F(\bar{\mathbf{w}}_T)\rangle \\
&\leq \frac{1}{T}\sum_{t=1}^{T} h(\mathbf{w}_t) - h(\mathbf{w}) + \frac{1}{T}\sum_{t=1}^{T}\langle \mathbf{w}_t - \mathbf{w}, F(\mathbf{w}_t)\rangle \\
&\leq \frac{L}{T} = O\left(\frac{1}{T}\right) ,
\end{aligned}$$

which establishes convergence rate for ADM. ■

The bound requires $\mathbf{x}$ and $\mathbf{y}$ to be bounded. In general, L is larger compard to the results in Theorem 6 and 7. According to (3.3),

$$\rho\sum_{t=0}^{T-1}(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) = \sum_{t=0}^{T-1}(\mathbf{y}_{t+1} - \mathbf{y}_t) = \mathbf{y}_T - \mathbf{y}_0 = \mathbf{y}_T , \tag{3.55}$$

meaning $\mathbf{y}_T$ is the sum of all past residuls of constraint violation and thus $\|\mathbf{y}\|_2$ is large. [80] also shows a similar result based on an auxiliary sequence $\{\mathbf{x}_{t+1}, \mathbf{z}_{t+1}, \tilde{\mathbf{y}}_{t+1} = \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{A}\mathbf{z}_t - \mathbf{c})\}$ instead of the sequence $\{\mathbf{x}_{t+1}, \mathbf{z}_{t+1}, \mathbf{y}_{t+1}\}$ generated by ADM. Compared to their proof, our proof is arguably simple and easier to understand. In fact, their proof is based on weak VI [143, 41, 56], while our proof is based on strong VI [143, 41, 56]. According to Minty's lemma [41, 56], they are equivalent if the solution set Ω is closed bounded and VI operator F is continuous and monotone.

Chapter 4

Bregman Alternating Direction Method of Multipliers

4.1 Introduction

In ADMM (3.3)-(3.5), the computational complexity of $\mathbf{y}$ update (3.5) is trivial. The computational complexity of ADMM lies in the $\mathbf{x}$ and $\mathbf{z}$ updates (3.3)-(3.4) which amount to solving proximal minimization problems using the quadratic penalty term. Inexact ADMM [214, 19] and generalized ADMM [48] have also been proposed to solve the updates inexactly by linearizing the functions and adding additional quadratic terms. Recently, online ADMM [200] and Bethe-ADMM [64] add an additional Bregman divergence on the $\mathbf{x}$ update by keeping or linearizing the quadratic penalty term $\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}-\mathbf{c}\|_2^2$. As far as we know, all existing ADMMs use quadratic penalty terms.

A large amount of literature shows that replacing the quadratic term by Bregman divergence in gradient-type methods could greatly boost their performance in solving the constrained optimization problem. First, the use of Bregman divergence could effectively exploit the structure of problems [33, 10, 52] , e.g., in computerized tomography [12], clustering problems and exponential family distributions [5]. Second, in some cases, the gradient descent method with Kullback-Leibler (KL) divergence can outperform the method with the quadratic term by a factor of $O(\sqrt{n \ln n})$ where n is the dimensionality of the problem [10, 12]. Mirror descent algorithm (MDA) and composite objective mirror descent (COMID) [52] use Bregman divergence to replace the quadratic term in gradient descent or proximal gradient [37]. Proximal point method with D-functions (PMD) [33, 25] and Bregman proximal minimization (BPM) [99] generalize proximal point method by using Bregman divegence to replace the quadratic term. On the side of ADMM, it is still unknown whether the quadratic penalty term in ADMM can be replaced by Bregman divergence, although the convergence of ADMM is well understood. However, as pointed out by [19], "There is currently no proof of convergence known for ADMM with nonquadratic penalty terms."

In this chapter, we propose Bregman ADMM (BADMM) which uses Bregman divergences to replace the quadratic penalty term in ADMM, answering the question raised in [19]. More specifically, the quadratic penalty term in the $\mathbf{x}$ and $\mathbf{z}$ updates (3.3)-(3.4) will be replaced by a Bregman divergence in BADMM. We also introduce a generalized version of BADMM where two additional Bregman divergences are added to the $\mathbf{x}$ and $\mathbf{z}$ updates. The generalized BADMM (BADMM for short) provides a unified framework for solving (3.1), which allows one to choose suitable Bregman divergence so that the $\mathbf{x}$ and $\mathbf{z}$ updates can be solved efficiently. BADMM includes ADMM and its variants as special cases. In particular, BADMM replaces all quadratic terms in generalized ADMM [48] with Bregman divergences. By choosing a proper Bregman divergence, we also show that inexact ADMM [214] and Bethe ADMM [64] can be considered as special cases of BADMM. BADMM generalizes ADMM similar to how MDA generalizes gradient descent and how PMD generalizes proximal methods. In BADMM, the $\mathbf{x}$ and $\mathbf{z}$ updates can take the form of MDA or PMD. We establish the global convergence and the $O(1/T)$ iteration complexity for BADMM. In some cases, we show that BADMM can outperform ADMM by a factor $O(n/\ln n)$. We evaluate the performance of BADMM in solving the linear program problem of mass transportation [82]. By exploiting the structure of the problem, BADMM leads to massive parallelism and can easily run on GPU. BADMM can even be orders of magnitude faster than highly optimized commercial software Gurobi. While Gurobi breaks down in solving a linear program of hundreds of millions of parameters in a server, BADMM takes hundreds of seconds running in a single GPU.

The rest of this chapter is organized as follows. In Section 4.2, we propose Bregman ADMM and discuss several special cases of BADMM. In Section 4.3, we establish the convergence of BADMM. In Section 4.4, we use BADMM to solve the mass transportation problem on a GPU.

4.2 Bregman Alternating Direction Method of Multipliers

Let $\phi : \Omega \rightarrow \mathbb{R}$ be a continuously differentiable and strictly convex function on the relative interior of a convex set Ω. Denote $\nabla\phi(\mathbf{y})$ as the gradient of ϕ at $\mathbf{y}$. We define Bregman divergence[1] $B_\phi : \Omega \times \text{ri}(\Omega) \rightarrow \mathbb{R}_+$ induced by ϕ as

$$B_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \nabla\phi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \ .$$

Since ϕ is convex, $B_\phi(\mathbf{x}, \mathbf{y}) \geq 0$ where the equality holds if and only if $\mathbf{x} = \mathbf{y}$. More details about Bregman divergence can be found in [33, 5]. Two of the most commonly used examples are squared Euclidean distance $B_\phi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2$ and KL divergence $B_\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n x_i \log \frac{x_i}{y_i}$.

Assuming $B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z})$ is well defined, we replace the quadratic penalty term in the augmented Lagrangian (3.2) by a Bregman divergence as follows:

$$L_\rho^\phi(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \rho B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}). \tag{4.1}$$

Unfortunately, we can not derive Bregman ADMM (BADMM) updates by simply solving $L_\rho^\phi(\mathbf{x}, \mathbf{z}, \mathbf{y})$ alternatingly as ADMM does because Bregman divergences are not necessarily convex in the second argument. More specifically, given $(\mathbf{z}_t, \mathbf{y}_t)$, $\mathbf{x}_{t+1}$ can be obtained by solving $\min_{\mathbf{x}} \ L_\rho^\phi(\mathbf{x}, \mathbf{z}_t, \mathbf{y}_t)$, where the quadratic penalty term $\frac{1}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$ for ADMM in (3.3) is replaced with $B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t)$ in the $\mathbf{x}$ update of BADMM. However, given $(\mathbf{x}_{t+1}, \mathbf{y}_t)$, we cannot obtain $\mathbf{z}_{t+1}$ by solving $\min_{\mathbf{z}} \ L_\rho^\phi(\mathbf{x}_{t+1}, \mathbf{z}, \mathbf{y}_t)$, since the term $B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z})$ need not be convex in $\mathbf{z}$. The observation motivates a closer look at the role of the quadratic term in ADMM.

In standard ADMM, the quadratic augmentation term added to the Lagrangian is just a penalty term to ensure the new updates do not violate the constraint significantly. Staying with these goals, we propose the $\mathbf{z}$ update augmentation term of BADMM to be: $B_\phi(\mathbf{B}\mathbf{z}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})$, instead of the quadratic penalty term $\frac{1}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c}\|_2^2$ in (3.3). Then, we get the following updates for BADMM:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x} \in \mathcal{X}}{\text{argmin}} \ f(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c} \rangle + \rho B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t) \ , \tag{4.2}$$

$$\mathbf{z}_{t+1} = \underset{\mathbf{z} \in \mathcal{Z}}{\text{argmin}} \ g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \rho B_\phi(\mathbf{B}\mathbf{z}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) \ , \tag{4.3}$$

[1] The definition of Bregman divergence has been generalized to nondifferentiable functions [99, 188].

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \ . \tag{4.4}$$

Compared to ADMM (3.3)-(3.5), BADMM simply uses a Bregman divergence to replace the quadratic penalty term in the $\mathbf{x}$ and $\mathbf{z}$ updates. It is worth noting that the same Bregman divergence B_ϕ is used in the $\mathbf{x}$ and $\mathbf{z}$ updates.

We consider a special case when $\mathbf{A} = -\mathbf{I}, \mathbf{B} = \mathbf{I}, \mathbf{c} = \mathbf{0}$. (4.2) is reduced to

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) + \langle \mathbf{y}_t, -\mathbf{x} + \mathbf{z}_t \rangle + \rho B_\phi(\mathbf{x}, \mathbf{z}_t) \ . \tag{4.5}$$

If ϕ is a quadratic function, the constrained problem (4.5) requires the projection onto the constraint set $\mathcal{X}$. However, in some cases, if choosing a proper Bregman divergence, (4.5) can be solved efficiently or has a closed-form solution. For example, if f is a linear function and $\mathcal{X}$ is the unit simplex, B_ϕ should be KL divergence, leading to the exponentiated gradient [10, 12, 144]. Interestingly, if the $\mathbf{z}$ update is also the exponentiated gradient, we have alternating exponentiated gradients. In Section 4, we will show the mass transportation problem can be cast into this scenario.

While the updates (4.2)-(4.3) use the same Bregman divergences, efficiently solving the $\mathbf{x}$ and $\mathbf{z}$ updates may not be feasible, especially when the structure of the original functions f, g, the function ϕ used for augmentation, and the constraint sets $\mathcal{X}, \mathcal{Z}$ are rather different. For example, if $f(\mathbf{x})$ is a logistic function in (4.5), it will not have a closed-form solution even B_ϕ is the KL divergence and $\mathcal{X}$ is the unit simplex. To address such concerns, we propose a generalized version of BADMM in Section 2.1.

4.2.1 Generalized BADMM

To allow the use of different Bregman divergences in the $\mathbf{x}$ and $\mathbf{z}$ updates (4.2)-(4.4) of BADMM, the generalized BADMM simply introduces an additional Bregman divergence for each update. The generalized BADMM has the following updates:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c} \rangle + \rho B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t) + \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}, \mathbf{x}_t) \ , \tag{4.6}$$

$$\mathbf{z}_{t+1} = \operatorname*{argmin}_{\mathbf{z} \in \mathcal{Z}} g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \rho B_\phi(\mathbf{B}\mathbf{z}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) + \rho_{\mathbf{z}} B_{\varphi_{\mathbf{z}}}(\mathbf{z}, \mathbf{z}_t) \ , \tag{4.7}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \tau(\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}) \ . \tag{4.8}$$

where $\rho > 0, \tau > 0, \rho_{\mathbf{x}} \geq 0, \rho_{\mathbf{z}} \geq 0$. Note that we allow the use of a different step size τ in the dual variable update [48, 86]. There are three Bregman divergences in the generalized BADMM. While the Bregman divergence B_ϕ is shared by the $\mathbf{x}$ and $\mathbf{z}$ updates, the $\mathbf{x}$ update has its own Bregman divergence $B_{\varphi_{\mathbf{x}}}$ and the $\mathbf{z}$ update has its own Bregman divergence $B_{\varphi_{\mathbf{z}}}$. The two additional Bregman divergences in generalized BADMM are variable specific, and can be chosen to make sure that the $\mathbf{x}_{t+1}, \mathbf{z}_{t+1}$ updates are efficient. If all three Bregman divergences are quadratic functions, the generalized BADMM reduces to the generalized ADMM [48]. We prove convergence of generalized BADMM in Section 3, which yields the convergence of BADMM with $\rho_x = \rho_z = 0$.

In the following, we illustrate how to choose a proper Bregman divergence $B_{\varphi_{\mathbf{x}}}$ so that the $\mathbf{x}$ update can be solved efficiently, e.g., a closed-form solution, noting that the same arguments apply to the $\mathbf{z}$-updates. Consider the first three terms in (7.21) as $s(\mathbf{x}) + h(\mathbf{x})$, where $s(\mathbf{x})$ denotes an easy term and $h(\mathbf{x})$ is the problematic term which needs to be linearized for an efficient $\mathbf{x}$-update. We illustrate the idea with several examples later in the section. Now, we have

$$\mathbf{x}_{t+1} = \min_{\mathbf{x}\in\mathcal{X}} \; s(\mathbf{x}) + h(\mathbf{x}) + \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}, \mathbf{x}_t) \; . \tag{4.9}$$

where efficient updates are difficult due to the mismatch in structure between h and $\mathcal{X}$. The goal is to 'linearize' the function h by using the fact that the Bregman divergence $B_h(\mathbf{x}, \mathbf{x}_t)$ captures all the higher-order (beyond linear) terms in $h(\mathbf{x})$ so that:

$$h(\mathbf{x}) - B_h(\mathbf{x}, \mathbf{x}_t) = h(\mathbf{x}_t) + \langle \mathbf{x} - \mathbf{x}_t, \nabla h(\mathbf{x}_t) \rangle \tag{4.10}$$

is a linear function of $\mathbf{x}$. Let ψ be another convex function such that one can efficiently solve $\min_{x\in\mathcal{X}} \; s(\mathbf{x}) + \psi(x) + \langle \mathbf{x}, \mathbf{b} \rangle$ for any constant $\mathbf{b}$. Assuming $\varphi_{\mathbf{x}}(\mathbf{x}) = \psi(\mathbf{x}) - \frac{1}{\eta} h(\mathbf{x})$ is convex, we construct a Bregman divergence based proximal term to the original problem so that:

$$\operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} s(\mathbf{x}) + h(\mathbf{x}) + \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}, \mathbf{x}_t) = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} s(\mathbf{x}) + \psi(\mathbf{x}) + \langle \mathbf{x}, \frac{1}{\rho_{\mathbf{x}}} \nabla h(\mathbf{x}_t) - \nabla \psi(\mathbf{x}_t) \rangle \; , \tag{4.11}$$

where the latter problem can be solved efficiently, by our assumption. To ensure $\varphi_{\mathbf{x}}$ is convex, we need the following condition:

Proposition 1 *If h is smooth and has Lipschitz continuous gradients with constant ν under a p-norm, then $\varphi_{\mathbf{x}}$ is $\nu/\rho_{\mathbf{x}}$-strongly convex w.r.t. the p-norm.*

This condition has been widely used in gradient-type methods, including MDA and COMID. Note that the convergence analysis of generalized ADMM in Section 4 holds for any additional Bregman divergence based proximal terms, and does not rely on such specific choices. Using the above idea, one can 'linearize' different parts of the $\mathbf{x}$ update to yield an efficient update.

We consider three special cases, respectively focusing on linearizing the function $f(\mathbf{x})$, linearizing the Bregman divergence based augmentation term $B_\phi(\mathbf{c}-\mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t)$, and linearizing both terms, along with examples for each case.

Case 1: Linearization of smooth function f: Let $h(\mathbf{x}) = f(\mathbf{x})$ in (4.11), we have

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \langle \mathbf{y}_t, \mathbf{A}\mathbf{x} \rangle + \rho B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t) + \rho_{\mathbf{x}} B_{\psi_{\mathbf{x}}}(\mathbf{x}, \mathbf{x}_t) \ . \tag{4.12}$$

where $\nabla f(\mathbf{x}_t)$ is the gradient of $f(\mathbf{x})$ at $\mathbf{x}_t$.

Example 1 Consider the following ADMM form for sparse logistic regression problem [74, 19]:

$$\min_{\mathbf{x}} h(\mathbf{x}) + \lambda \|\mathbf{z}\|_1 \ , \text{ s.t. } \mathbf{x} = \mathbf{z} \ , \tag{4.13}$$

where $h(\mathbf{x})$ is the logistic function. If we use ADMM to solve (4.13), the $\mathbf{x}$ update is as follows [19]:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \ h(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{x} - \mathbf{z}_t \rangle + \frac{\rho}{2} \|\mathbf{x} - \mathbf{z}_t\|_2^2 \ , \tag{4.14}$$

which is a ridge-regularized logistic regression problem and one needs an iterative algorithm like L-BFGS to solve it. Instead, if we linearize $h(\mathbf{x})$ at $\mathbf{x}_t$ and set B_ψ to be a quadratic function, then

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}} \ \langle \nabla \ h(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \langle \mathbf{y}_t, \mathbf{x} - \mathbf{z}_t \rangle + \frac{\rho}{2} \|\mathbf{x} - \mathbf{z}_t\|_2^2 + \frac{\rho_{\mathbf{x}}}{2} \|\mathbf{x} - \mathbf{x}_t\|_2^2 \ , \tag{4.15}$$

the $\mathbf{x}$ update has a simple closed-form solution.

Case 2: Linearization of the quadratic penalty term: In ADMM, $B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}, \mathbf{B}\mathbf{z}_t) = \frac{1}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$. Let $h(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$. Then $\nabla h(\mathbf{x}_t) = \mathbf{A}^T(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c})$, we have

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \ f(\mathbf{x}) + \langle \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c}), \mathbf{A}\mathbf{x} \rangle + \rho_{\mathbf{x}} B_\psi(\mathbf{x}, \mathbf{x}_t) \ . \tag{4.16}$$

The case mainly solves the problem due to the $\mathbf{Ax}$ term which makes $\mathbf{x}$ updates nonseparable, whereas the linearized version can be solved with separable (parallel) updates. Several problems have been benefited from the linearization of quadratic term [48], e.g., when f is ℓ_1 loss function [74], and projection onto the unit simplex or ℓ_1 ball [51].

Case 3: Mirror Descent: In some settings, we want to linearize both the function f and the quadratic augmentation term $B_\phi(\mathbf{c} - \mathbf{Ax}, \mathbf{Bz}_t) = \frac{1}{2}\|\mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\|_2^2$. Let $h(\mathbf{x}) = f(\mathbf{x}) + \langle \mathbf{y}^t, \mathbf{Ax}\rangle + \frac{\rho}{2}\|\mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\|_2^2$, we have

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}} \langle \nabla h(\mathbf{x}_t), \mathbf{x}\rangle + \rho_{\mathbf{x}} B_\psi(\mathbf{x}, \mathbf{x}_t) \ . \tag{4.17}$$

Note that (6.63) is a MDA-type update. Further, one can do a similar exercise with a general Bregman divergence based augmentation term $B_\phi(\mathbf{c} - \mathbf{Ax}, \mathbf{Bz}_t)$, although there has to be a good motivation for going to this route.

Example 2 [Bethe-ADMM [64]] Given an undirected graph $G = (V, E)$, where V is the vertex set and E is the edge set. Assume a random discrete variable X_i associated with node $i \in V$ can take K values. In a pairwise MRF, the joint distribution of a set of discrete random variables $X = \{X_1, \cdots, X_n\}$ (n is the number of nodes in the graph) is defined in terms of nodes and cliques [198]. Consider solving the following graph-structured problem :

$$\min\ l(\boldsymbol{\mu}) \ \text{ s.t. } \ \boldsymbol{\mu} \in \mathbb{L}(G) \ , \tag{4.18}$$

where $l(\boldsymbol{\mu})$ is a decomposable function of $\boldsymbol{\mu}$ and $\mathbb{L}(G)$ is the so-called local polytope [198] determined by the marginalization and normalization (MN) constraints for each node and edge in the graph G:

$$\mathbb{L}(G) = \{\boldsymbol{\mu} \geq 0 \ , \sum_{x_i} \mu_i(x_i) = 1 \ , \sum_{x_j} \mu_{ij}(x_i, x_j) = \mu_i(x_i)\} \ , \tag{4.19}$$

where μ_i, μ_{ij} are pseudo-marginal distributions of node i and edge ij respectively. In particular, (4.18) serves as a LP relaxation of MAP inference probem in a pairwise MRF if $l(\boldsymbol{\mu})$ is defined as follows:

$$l(\boldsymbol{\mu}) = \sum_i \sum_{x_i} \theta_i(x_i)\mu_i(x_i) + \sum_{ij\in E} \sum_{x_{ij}} \theta_{ij}(x_i, x_j)\mu_{ij}(x_i, x_j), \tag{4.20}$$

where θ_i, θ_{ij} are the potential functions of node i and edge ij respectively.

The complexity of polytope $\mathbb{L}(G)$ makes (4.18) difficult to solve. One possible way is to decompose the graph into trees such that

$$\min \sum_{\tau} c_\tau l_\tau(\boldsymbol{\mu}_\tau) \text{ s.t. } \boldsymbol{\mu}_\tau \in \mathbb{T}_\tau, \boldsymbol{\mu}_\tau = \mathbf{m}_\tau \;, \tag{4.21}$$

where $\mathbb{T}_\tau$ denotes the MN constraints (4.19) in the tree τ. $\boldsymbol{\mu}_\tau$ is a vector of pseudo-marginals of nodes and edges in the tree τ. $\mathbf{m}$ is a global variable which contains all trees and $\mathbf{m}_\tau$ corresponds to the tree τ in the global variable. c_τ is the weight for sharing variables. The augmented Lagrangian is

$$L_\rho(\boldsymbol{\mu}_\tau, \mathbf{m}, \boldsymbol{\lambda}_\tau) = \sum_{\tau} c_\tau l_\tau(\boldsymbol{\mu}_\tau) + \langle \boldsymbol{\lambda}_\tau, \boldsymbol{\mu}_\tau - \mathbf{m}_\tau \rangle + \frac{\rho}{2}\|\boldsymbol{\mu}_\tau - \mathbf{m}_\tau\|_2^2 \;. \tag{4.22}$$

which leads to the following update for $\boldsymbol{\mu}_\tau^{t+1}$ in ADMM:

$$\boldsymbol{\mu}_\tau^{t+1} = \underset{\boldsymbol{\mu}_\tau \in \mathbb{T}_\tau}{\operatorname{argmin}}\; c_\tau l_\tau(\boldsymbol{\mu}_\tau) + \langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau \rangle + \frac{\rho}{2}\|\boldsymbol{\mu}_\tau - \mathbf{m}_\tau^t\|_2^2 \tag{4.23}$$

(4.23) is difficult to solve due to the MN constraints in the tree. Let $h(\boldsymbol{\mu}_\tau)$ be the objective of (4.23). If linearizing $h(\boldsymbol{\mu}_\tau)$ and adding a Bregman divergence in (4.23), we have:

$$\begin{aligned}
\boldsymbol{\mu}_\tau^{t+1} &= \underset{\boldsymbol{\mu}_\tau \in \mathbb{T}_\tau}{\operatorname{argmin}}\; \langle \nabla h(\boldsymbol{\mu}_\tau^t), \boldsymbol{\mu}_\tau \rangle + \rho_{\mathbf{x}} B_\psi(\boldsymbol{\mu}_\tau, \boldsymbol{\mu}_\tau^t) \\
&= \underset{\boldsymbol{\mu}_\tau \in \mathbb{T}_\tau}{\operatorname{argmin}}\; \langle \nabla h(\boldsymbol{\mu}_\tau^t) - \rho_{\mathbf{x}} \nabla \psi(\boldsymbol{\mu}_\tau^t), \boldsymbol{\mu}_\tau \rangle + \rho_{\mathbf{x}} \psi(\boldsymbol{\mu}_\tau) \;,
\end{aligned}$$

If $\psi(\boldsymbol{\mu}_\tau)$ is the negative Bethe entropy of $\boldsymbol{\mu}_\tau$, the update of $\boldsymbol{\mu}_\tau^{t+1}$ becomes the Bethe entropy problem [198] and can be solved exactly by the sum-product algorithm in a linear time in the tree.

4.3 Convergence Analysis of BADMM

We need the following assumption in establishing the convergence of BADMM:

Assumption 6

(a) $f : \mathbb{R}^{n_1} \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^{n_2} \to \mathbb{R} \cup \{+\infty\}$ are closed, proper and convex.

(b) An optimal solution exists.

(c) The Bregman divergence B_ϕ is defined on an α-strongly convex function ϕ with respect to a p-norm $\|\cdot\|_p^2$, i.e., $B_\phi(\mathbf{u}, \mathbf{v}) \geq \frac{\alpha}{2}\|\mathbf{u} - \mathbf{v}\|_p^2$, where $\alpha > 0$.

We start wth the Lagrangian, which is defined as follows:

$$L(\mathbf{x},\mathbf{y},\mathbf{z}) = f(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}\rangle. \tag{4.24}$$

Assume that $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ satisfies the KKT conditions of (6.5), i.e.,

$$-\mathbf{A}^T\mathbf{y}^* \in \partial f(\mathbf{x}^*) \ , \tag{4.25}$$

$$-\mathbf{B}^T\mathbf{y}^* \in \partial g(\mathbf{z}^*) \ , \tag{4.26}$$

$$\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* - \mathbf{c} = 0 \ . \tag{4.27}$$

$\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ is an optimal solution. The optimality conditions of (7.21) and (4.7) are

$$-\mathbf{A}^T\{\mathbf{y}_t + \rho(-\nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) + \nabla\phi(\mathbf{B}\mathbf{z}_t)\} - \rho_{\mathbf{x}}(\nabla\varphi_{\mathbf{x}}(\mathbf{x}_{t+1}) - \nabla\varphi_{\mathbf{x}}(\mathbf{x}_t)) \in \partial f(\mathbf{x}_{t+1}) \ , \tag{4.28}$$

$$-\mathbf{B}^T\{\mathbf{y}_t + \rho(\nabla\phi(\mathbf{B}\mathbf{z}_{t+1}) - \nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})\} - \rho_{\mathbf{z}}(\nabla\varphi_{\mathbf{z}}(\mathbf{z}_{t+1}) - \nabla\varphi_{\mathbf{z}}(\mathbf{z}_t)) \in \partial g(\mathbf{z}_{t+1}) \ . \tag{4.29}$$

If $\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} = \mathbf{c}$, then $\mathbf{y}_{t+1} = \mathbf{y}_t$. Therefore, (4.25) is satisfied if $\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t = \mathbf{c}\ , \mathbf{x}_{t+1} = \mathbf{x}_t$ in (4.28). Similarly, (4.26) is satisfied if $\mathbf{z}_{t+1} = \mathbf{z}_t$ in (4.29). Overall, the KKT conditions (4.25)-(4.27) are satisfied if the following optimality conditions are satisfied:

$$B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t) = 0 \ , \ B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t) = 0 \ , \tag{4.30a}$$

$$\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c} = 0 \ , \ \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} = 0 \ . \tag{4.30b}$$

For the exact BADMM, $\rho_{\mathbf{x}} = \rho_{\mathbf{z}} = 0$ in (7.21) and (4.7), the optimality conditions are (4.30b), which is equivalent to the optimality conditions used in the proof of ADMM in [19], i.e.,

$$\mathbf{B}\mathbf{z}_{t+1} - \mathbf{B}\mathbf{z}_t = 0 \ , \ \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} = 0 \ . \tag{4.31}$$

Define the residuals of optimality conditions (4.30) at $(t+1)$ as:

$$\begin{aligned} R(t+1) = {} & B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t) + \gamma\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \\ & + \frac{\rho_{\mathbf{x}}}{\rho} B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t) + \frac{\rho_{\mathbf{z}}}{\rho} B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t) \ , \end{aligned} \tag{4.32}$$

where $\gamma > 0$. If $R(t+1) = 0$, the optimality conditions (4.30) and (4.30b) are satisfied. It is sufficient to show the convergence of BADMM by showing $R(t+1)$ converges to zero. We need the following lemma.

Lemma 11 *Let the sequence $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by Bregman ADMM (7.21)-(4.8). For any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, we have*

$$\begin{aligned}
& f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \\
& \leq -\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle - \rho(B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t) + B_\phi(\mathbf{B}\mathbf{z}_{t+1}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})) \\
& + \rho(B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) - B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_{t+1})) + \rho_{\mathbf{x}}(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t)) \\
& + \rho_{\mathbf{z}}(B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_t) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_{t+1}) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t)) \ . \qquad (4.33)
\end{aligned}$$

Proof: Using the convexity of f and its subgradient given in (4.28), we have

$$\begin{aligned}
& f(\mathbf{x}_{t+1}) - f(\mathbf{x}) \\
& \leq \langle -\mathbf{A}^T\{\mathbf{y}_t + \rho(-\nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) + \nabla\phi(\mathbf{B}\mathbf{z}_t)\} - \rho_{\mathbf{x}}(\nabla\varphi_{\mathbf{x}}(\mathbf{x}_{t+1}) - \nabla\varphi_{\mathbf{x}}(\mathbf{x}_t)), \mathbf{x}_{t+1} - \mathbf{x}\rangle \\
& = -\langle \mathbf{y}_t, \mathbf{A}(\mathbf{x}_{t+1} - \mathbf{x})\rangle + \rho\langle \nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{B}\mathbf{z}_t), \mathbf{A}(\mathbf{x}_{t+1} - \mathbf{x})\rangle \\
& - \rho_{\mathbf{x}}\langle \nabla\varphi_{\mathbf{x}}(\mathbf{x}_{t+1}) - \nabla\varphi_{\mathbf{x}}(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}\rangle \ . \qquad (4.34)
\end{aligned}$$

Setting $\mathbf{x} = \mathbf{x}^*$ and using $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, we have

$$\begin{aligned}
& f(\mathbf{x}_{t+1}) - f(\mathbf{x}^*) \\
& \leq -\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}^* - \mathbf{c}\rangle + \rho\langle \nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{B}\mathbf{z}_t), \mathbf{B}\mathbf{z}^* - (\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})\rangle \\
& - \rho_{\mathbf{x}}\langle \nabla\varphi_{\mathbf{x}}(\mathbf{x}_{t+1}) - \nabla\varphi_{\mathbf{x}}(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}\rangle \\
& = -\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}^* - \mathbf{c}\rangle + \rho(B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) - B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t)) \\
& + \rho_{\mathbf{x}}(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t)) \ . \qquad (4.35)
\end{aligned}$$

where the last equality uses the three point property of Bregman divergence, i.e.,

$$\langle \nabla\phi(\mathbf{u}) - \nabla\phi(\mathbf{v}), \mathbf{w} - \mathbf{u}\rangle = B_\phi(\mathbf{w}, \mathbf{v}) - B_\phi(\mathbf{w}, \mathbf{u}) - B_\phi(\mathbf{u}, \mathbf{v}) \ . \qquad (4.36)$$

Similarly, using the convexity of g and its subgradient given in (4.29), for any $\mathbf{z}$,

$$\begin{aligned}
& g(\mathbf{z}_{t+1}) - g(\mathbf{z}) \\
& \leq \langle -\mathbf{B}^T\{\mathbf{y}_t + \rho(\nabla\phi(\mathbf{B}\mathbf{z}_{t+1}) - \nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})\} - \rho_{\mathbf{z}}(\nabla\varphi_{\mathbf{z}}(\mathbf{z}_{t+1}) - \nabla\varphi_{\mathbf{z}}(\mathbf{z}_t)), \mathbf{z}_{t+1} - \mathbf{z}\rangle \\
& = -\langle \mathbf{y}_t, \mathbf{B}(\mathbf{z}_{t+1} - \mathbf{z})\rangle + \rho\langle \nabla\phi(\mathbf{B}\mathbf{z}_{t+1}) - \nabla\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}), \mathbf{B}\mathbf{z} - \mathbf{B}\mathbf{z}_{t+1})\rangle \\
& - \rho_{\mathbf{z}}\langle \nabla\varphi_{\mathbf{z}}(\mathbf{z}_{t+1}) - \nabla\varphi_{\mathbf{z}}(\mathbf{z}_t), \mathbf{z}_{t+1} - \mathbf{z}\rangle \\
& = -\langle \mathbf{y}_t, \mathbf{B}(\mathbf{z}_{t+1} - \mathbf{z})\rangle + \rho\left\{B_\phi(\mathbf{B}\mathbf{z}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - B_\phi(\mathbf{B}\mathbf{z}, \mathbf{B}\mathbf{z}_{t+1}) - B_\phi(\mathbf{B}\mathbf{z}_{t+1}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})\right\}
\end{aligned}$$

$$+ \rho_{\mathbf{z}}(B_{\varphi_{\mathbf{z}}}(\mathbf{z}, \mathbf{z}_t) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}, \mathbf{z}_{t+1}) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t)) \ . \tag{4.37}$$

where the last equality uses the three point property of Bregman divergence (4.36). Set $\mathbf{z} = \mathbf{z}^*$ in (4.37). Adding (4.35) and (4.37) completes the proof. ■

Under Assumption 6(c), the following lemma shows that (4.32) is bounded by a telescoping series of $D(\mathbf{w}^*, \mathbf{w}_t) - D(\mathbf{w}^*, \mathbf{w}_{t+1})$, where $D(\mathbf{w}^*, \mathbf{w}_t)$ defines the distance from the current iterate $\mathbf{w}_t = (\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t)$ to a KKT point $\mathbf{w}^* = (\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ as follows:

$$D(\mathbf{w}^*, \mathbf{w}_t) = \frac{1}{2\tau\rho}\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 + B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) + \frac{\rho_{\mathbf{x}}}{\rho} B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) + \frac{\rho_{\mathbf{z}}}{\rho} B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_t) \ . \tag{4.38}$$

Lemma 12 *Let the sequence $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by Bregman ADMM (7.21)-(4.8) and $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ satisfying (4.25)-(4.27). Let the Assumption 6 hold. $R(t+1)$ and $D(\mathbf{w}^*, \mathbf{w}_t)$ are defined in (4.32) and (4.38) respectively. Set $\tau \leq (\alpha\sigma - 2\gamma)\rho$, where $\sigma = \min\{1, m^{\frac{2}{p}-1}\}$ and $0 < \gamma < \frac{\alpha\sigma}{2}$. Then*

$$R(t+1) \leq D(\mathbf{w}^*, \mathbf{w}_t) - D(\mathbf{w}^*, \mathbf{w}_{t+1}) \ . \tag{4.39}$$

Proof: Assume $\{\mathbf{x}^*, \mathbf{y}^*\}$ satisfies (4.25). Since f is convex, then

$$f(\mathbf{x}^*) - f(\mathbf{x}_{t+1}) \leq -\langle \mathbf{A}^T\mathbf{y}^*, \mathbf{x}^* - \mathbf{x}_{t+1}\rangle = -\langle \mathbf{y}^*, \mathbf{A}\mathbf{x}^* - \mathbf{A}\mathbf{x}_{t+1}\rangle \ . \tag{4.40}$$

Similarly, for convex function g and $\{\mathbf{z}^*, \mathbf{y}^*\}$ satisfying (4.26), we have

$$g(\mathbf{z}^*) - g(\mathbf{z}_{t+1}) \leq -\langle \mathbf{B}^T\mathbf{y}^*, \mathbf{z}^* - \mathbf{z}_{t+1}\rangle = -\langle \mathbf{y}^*, \mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\rangle \ . \tag{4.41}$$

Adding them together and using the fact that $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, we have

$$f(\mathbf{x}^*) + g(\mathbf{z}^*) - (f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1})) \leq \langle \mathbf{y}^*, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle \ . \tag{4.42}$$

Adding (4.42) and (4.33) together yields

$$\begin{aligned}
&0 \leq \langle \mathbf{y}^* - \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle - \rho(B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t) + B_\phi(\mathbf{B}\mathbf{z}_{t+1}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1})) \\
&+ \rho(B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) - B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_{t+1})) + \rho_{\mathbf{x}}(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t)) \\
&+ \rho_{\mathbf{z}}(B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_t) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_{t+1}) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t)) \ .
\end{aligned} \tag{4.43}$$

Using $\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} = \frac{1}{\tau}(\mathbf{y}_{t+1} - \mathbf{y}_t)$, the first term can be rewritten as

$$\langle \mathbf{y}^* - \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle = \frac{1}{\tau}\langle \mathbf{y}^* - \mathbf{y}_t, \mathbf{y}_{t+1} - \mathbf{y}_t\rangle$$

$$
\begin{aligned}
&= \frac{1}{2\tau}\left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2 + \|\mathbf{y}_{t+1} - \mathbf{y}_t\|_2^2\right) \\
&= \frac{1}{2\tau}\left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2\right) + \frac{\tau}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \, . \qquad (4.44)
\end{aligned}
$$

Plugging into (4.43) and rearranging the terms, we have

$$
\begin{aligned}
&\frac{1}{2\tau}\left(\|\mathbf{y}^* - \mathbf{y}_t\|_2^2 - \|\mathbf{y}^* - \mathbf{y}_{t+1}\|_2^2\right) + \rho(B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) - B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_{t+1})) \\
&\rho_{\mathbf{x}}(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_{t+1})) + \rho_{\mathbf{z}}(B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_t) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_{t+1})) \\
&\geq \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t) + \rho_{\mathbf{z}} B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t) + \rho B_\phi(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t) \\
&+ \rho B_\phi(\mathbf{B}\mathbf{z}_{t+1}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - \frac{\tau}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \, . \qquad (4.45)
\end{aligned}
$$

Dividing both sides by ρ and letting $R(t+1)$ and $D(\mathbf{w}^*, \mathbf{w}_t)$ be defined in (4.32) and (4.38) respectively, we have

$$
\begin{aligned}
&D(\mathbf{w}^*, \mathbf{w}_t) - D(\mathbf{w}^*, \mathbf{w}_{t+1}) \\
&\geq R(t+1) + B_\phi(\mathbf{B}\mathbf{z}_{t+1}, \mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}) - (\frac{\tau}{2\rho} + \gamma)\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \\
&\qquad \geq R(t+1) + \frac{\alpha}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_p^2 - (\frac{\tau}{2\rho} + \gamma)\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \, ,
\end{aligned}
$$
(4.46)

where the last inequality uses the Assumption 6(c).

If $0 < p \leq 2$, $\|\mathbf{u}\|_p \geq \|\mathbf{u}\|_2$. Set $\frac{\alpha}{2} \geq \frac{\tau}{2\rho} + \gamma$ in (4.46), i.e., $\tau \leq (\alpha - 2\gamma)\rho$. We can always find a $\gamma < \frac{\alpha}{2}$, thus (4.39) follows.

If $p > 2$, $\|\mathbf{u}\|_2 \leq m^{\frac{1}{2}-\frac{1}{p}}\|\mathbf{u}\|_p$ for any $\mathbf{u} \in \mathbb{R}^{m\times 1}$, so $\|\mathbf{u}\|_p^2 \geq m^{\frac{2}{p}-1}\|\mathbf{u}\|_2^2$. In (4.46), set $\frac{\alpha}{2}m^{\frac{2}{p}-1} \geq \frac{\tau}{2\rho} + \gamma$, i.e., $\tau \leq (\alpha m^{\frac{2}{p}-1} - 2\gamma)\rho$. As long as $\gamma < \frac{\alpha}{2}m^{\frac{2}{p}-1}$, we have (4.39). ■

Remark 1 (a) If $0 < p \leq 2$, then $\sigma = 1$ and $\tau \leq (\alpha - 2\gamma)\rho$. The case that $0 < p \leq 2$ includes two widely used Bregman divergences, i.e., Euclidean distance and KL divergence. For KL divergence in the unit simplex, we have $\alpha = 1, p = 1$ in the Assumption 6 (c), i.e., $KL(\mathbf{u}, \mathbf{v}) \geq \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|_1^2$ [10].

(b) Since we often set B_ϕ to be a quadratic function ($p = 2$), the three special cases in Section 2.1 could choose step size $\tau = (\alpha - 2\gamma)\rho$.

(c) If $p > 2$, the proof requires a sufficiently small step size τ, which may not be needed in practice. It would be interesting to see whether we can use a same $\tau = O(\rho)$ for any $p > 0$ using other proof techniques.

The following theorem establishes the global convergence for BADMM.

Theorem 9 *Let the sequence $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by Bregman ADMM (7.21)-(4.8) and $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ satisfying (4.25)-(4.27). Let the Assumption 6 hold and τ, γ satisfy the conditions in Lemma 12. Then $R(t+1)$ converges to zero and $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ converges to a KKT point $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ of (6.5).*

Proof: Since $R(t+1) \geq 0$, (4.39) implies $D(\mathbf{w}^*, \mathbf{w}_{t+1}) \leq D(\mathbf{w}^*, \mathbf{w}_t)$. Therefore, $D(\mathbf{w}^*, \mathbf{w}_t)$ is monotonically nonincreasing and $\mathbf{w}_t$ converges to a KKT point $\mathbf{w}^*$. Summing (4.39) over t from 0 to ∞ yields

$$\sum_{t=0}^{\infty} R(t+1) \leq D(\mathbf{w}^*, \mathbf{w}_0) \ . \tag{4.47}$$

Since $R(t+1) \geq 0$, $R(t+1) \to 0$ as $t \to \infty$, which completes the proof. ■

The following theorem establishs a $O(1/T)$ convergence rate for the objective and residual of constraints in an ergodic sense.

Theorem 10 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by Bregman ADMM (7.21),(4.7),(4.8) and $\mathbf{y}_0 = \mathbf{0}$. Let $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$, $\bar{\mathbf{z}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{z}_t$. Set $\tau \leq (\alpha\sigma - 2\gamma)\rho$, where $\sigma = \min\{1, m^{\frac{2}{p}-1}\}$ and $0 < \gamma < \frac{\alpha\sigma}{2}$. For any $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ satisfying KKT conditions (4.25)-(4.27), we have*

$$f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \frac{D_1}{T} \ , \tag{4.48}$$

$$\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T - \mathbf{c}\|_2^2 \leq \frac{D(\mathbf{w}^*, \mathbf{w}_0)}{\gamma T} \ , \tag{4.49}$$

where $D_1 = \rho B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_0) + \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_0) + \rho_{\mathbf{z}} B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_0)$.

Proof: Using (4.8), we have

$$\begin{aligned} -\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c} \rangle &= -\frac{1}{\tau}\langle \mathbf{y}_t, \mathbf{y}_{t+1} - \mathbf{y}_t \rangle \\ &= -\frac{1}{2\tau}(\|\mathbf{y}_{t+1}\|_2^2 - \|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1} - \mathbf{y}_t\|_2^2) \\ &= \frac{1}{2\tau}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{\tau}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \ . \end{aligned} \tag{4.50}$$

Plugging into (4.33) and ignoring some negative terms yield

$$f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \frac{1}{2\tau}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2)$$

$$+ \rho(B_\phi(\mathbf{Bz}^*, \mathbf{Bz}_t) - B_\phi(\mathbf{Bz}^*, \mathbf{Bz}_{t+1})) + \rho_\mathbf{x}(B_{\varphi_\mathbf{x}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_\mathbf{x}}(\mathbf{x}^*, \mathbf{x}_{t+1}))$$
$$+ \rho_\mathbf{z}(B_{\varphi_\mathbf{z}}(\mathbf{z}^*, \mathbf{z}_t) - B_{\varphi_\mathbf{z}}(\mathbf{z}^*, \mathbf{z}_{t+1})) - \rho B_\phi(\mathbf{Bz}_{t+1}, \mathbf{c} - \mathbf{Ax}_{t+1}) + \frac{\tau}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 \,. \tag{4.51}$$

Assume $B_\phi(\mathbf{Bz}_{t+1}, \mathbf{c}-\mathbf{Ax}_{t+1}) \geq \frac{\alpha}{2}\|\mathbf{Ax}_{t+1}+\mathbf{Bz}_{t+1}-\mathbf{c}\|_p^2$. If $0 < p \leq 2$, using $\|\mathbf{u}\|_p \leq \|\mathbf{u}\|_2$,

$$-\rho B_\phi(\mathbf{Bz}_{t+1}, \mathbf{c} - \mathbf{Ax}_{t+1}) + \frac{\tau}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 \leq -\frac{\alpha\rho - \tau}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 \,.$$

Setting $\tau \leq (\alpha - 2\gamma)\rho$, the last two terms on the right hand side of (4.51) can be removed.

If $p > 2$, $\|\mathbf{u}\|_2 \leq m^{\frac{1}{2}-\frac{1}{p}}\|\mathbf{u}\|_p$ for any $\mathbf{u} \in \mathbb{R}^{m\times 1}$, so $\|\mathbf{u}\|_p^2 \geq m^{\frac{2}{p}-1}\|\mathbf{u}\|_2^2$. Then

$$-\rho B_\phi(\mathbf{Bz}_{t+1}, \mathbf{c} - \mathbf{Ax}_{t+1}) + \frac{\tau}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 \leq -\frac{\alpha\rho m^{\frac{2}{p}-1} - \tau}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 \,.$$

Setting $\tau \leq (\alpha m^{\frac{2}{p}-1} - 2\gamma)\rho$, the last two terms on the right hand side of (4.51) can be removed. Summing over t from 0 to $T-1$, we have the following telescoping sum

$$\begin{aligned}
&\sum_{t=0}^{T-1}[f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))] \\
&\leq \frac{1}{2\tau}\|\mathbf{y}_0\|_2^2 + \rho B_\phi(\mathbf{Bz}^*, \mathbf{Bz}_0) + \rho_\mathbf{x} B_{\varphi_\mathbf{x}}(\mathbf{x}^*, \mathbf{x}_0) + \rho_\mathbf{z}(B_{\varphi_\mathbf{z}}(\mathbf{z}^*, \mathbf{z}_0) \\
&= \rho B_\phi(\mathbf{Bz}^*, \mathbf{Bz}_0) + \rho_\mathbf{x} B_{\varphi_\mathbf{x}}(\mathbf{x}^*, \mathbf{x}_0) + \rho_\mathbf{z}(B_{\varphi_\mathbf{z}}(\mathbf{z}^*, \mathbf{z}_0) \,.
\end{aligned} \tag{4.52}$$

Dividing both sides by T and applying the Jensen's inequality gives (4.54).

Dividing both sides of (4.47) by T and applying the Jensen's inequality yield (4.55). ■

We consider one special case of BADMM which could outperform ADMM. Assume $\mathbf{B} = \mathbf{I}$ and $\mathcal{X}, \mathcal{Z}$ are the unit simplex. Let B_ϕ be the KL divergence. For $\mathbf{z} \in \mathbb{R}^{n_2\times 1}$, we have

$$B_\phi(\mathbf{z}^*, \mathbf{z}_0) = \sum_{i=1}^{n_2} z_i^* \ln\frac{z_i^*}{z_{i,0}} = \sum_{i=1}^{n_2} z_i^* \ln z_i^* + \ln n_2 \leq \ln n_2 \,. \tag{4.53}$$

Similarly, if $\rho_\mathbf{x} > 0$, by choosing $\mathbf{x}_0 = \mathbf{e}/n_2$, $B_{\varphi_\mathbf{x}}(\mathbf{x}^*, \mathbf{x}_0) \leq \ln n_1$. Setting $\alpha = 1, \sigma = 1$ and $\gamma = \frac{1}{4}$ in Theorem 10 yields the following result:

Corollary 1 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by Bregman ADMM (7.21),(4.7),(4.8) and $\mathbf{y}_0 = \mathbf{0}$. Assume $\mathbf{B} = \mathbf{I}$, and $\mathcal{X}$ and $\mathcal{Z}$ is the unit simplex. Let $B_\phi, B_{\varphi_\mathbf{x}}, B_{\varphi_\mathbf{z}}$ be KL* *divergence. Let* $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t$*,* $\bar{\mathbf{z}}_T = \frac{1}{T}\sum_{t=1}^{T}\mathbf{z}_t$*. Set* $\tau = \frac{3\rho}{4}$*. For any* $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ *satisfying KKT conditions (4.25)-(4.27), we have*

$$f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \frac{\rho \ln n_2 + \rho_{\mathbf{x}} \ln n_1 + \rho_{\mathbf{z}} \ln n_2}{T} , \tag{4.54}$$

$$\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T - \mathbf{c}\|_2^2 \leq \frac{\frac{2}{\tau\rho}\|\mathbf{y}^* - \mathbf{y}_0\|_2^2 + 4\ln n_2 + \frac{4\rho_{\mathbf{x}}}{\rho}\ln n_1 + \frac{4\rho_{\mathbf{z}}}{\rho}\ln n_2}{T} , \tag{4.55}$$

Remark 2 (a) In [10], it shows that MDA yields a smilar $O(\ln n)$ bound where n is dimensionality of the problem. If the diminishing step size of MDA is propotional to $\sqrt{\ln n}$, the bound is $O(\sqrt{\ln n})$. Therefore, MDA can outperform the gradient method by a factor $O((n/\ln n)^{1/2})$.

(b) With constant step size, BADMM outperforms ADMM by a factor $O(n/\ln n)$ in an ergodic sense.

4.4 Experimental Results

In this section, we use BADMM to solve the mass transportation problem [82]:

$$\min \quad \langle \mathbf{C}, \mathbf{X} \rangle \quad \text{s.t.} \quad \mathbf{X}\mathbf{e} = \mathbf{a}, \mathbf{X}^T\mathbf{e} = \mathbf{b}, \mathbf{X} \geq 0 \ . \tag{4.56}$$

where $\langle \mathbf{C}, \mathbf{X} \rangle$ denotes $\text{Tr}(\mathbf{C}^T\mathbf{X})$, $\mathbf{C} \in \mathbb{R}^{m \times n}$ is a cost matrix, $\mathbf{e}$ is a column vector of ones. (4.56) is called the assignment problem and can be solved exactly by the Hungarian method [108]. The mass transportation problem (4.56) is a linear program and thus can be solved by the simplex method.

We now show that (4.56) can be solved by ADMM and BADMM. We first introduce a variable $\mathbf{Z}$ to split the constraints into two simplex such that $\mathbf{\Delta}_{\mathbf{x}} = \{\mathbf{X} | \mathbf{X} \geq 0, \mathbf{X}\mathbf{e} = \mathbf{a}\}$ and $\mathbf{\Delta}_{\mathbf{z}} = \{\mathbf{Z} | \mathbf{Z} \geq 0, \mathbf{Z}^T\mathbf{e} = \mathbf{b}\}$. (4.56) can be rewritten in the following ADMM form:

$$\min \quad \langle \mathbf{C}, \mathbf{X} \rangle \quad \text{s.t.} \quad \mathbf{X} \in \mathbf{\Delta}_{\mathbf{x}}, \mathbf{Z} \in \mathbf{\Delta}_{\mathbf{z}}, \mathbf{X} = \mathbf{Z} \ . \tag{4.57}$$

(4.57) can be solved by ADMM which requires the Euclidean projection onto the simplex $\mathbf{\Delta}_{\mathbf{x}}$ and $\mathbf{\Delta}_{\mathbf{z}}$, although the projection can be done efficiently [51]. We use BADMM to solve (4.57):

$$\mathbf{X}^{t+1} = \underset{\mathbf{X} \in \mathbf{\Delta}_{\mathbf{x}}}{\text{argmin}} \langle \mathbf{C}, \mathbf{X} \rangle + \langle \mathbf{Y}^t, \mathbf{X} \rangle + \rho \text{KL}(\mathbf{X}, \mathbf{Z}^t) \ , \tag{4.58}$$

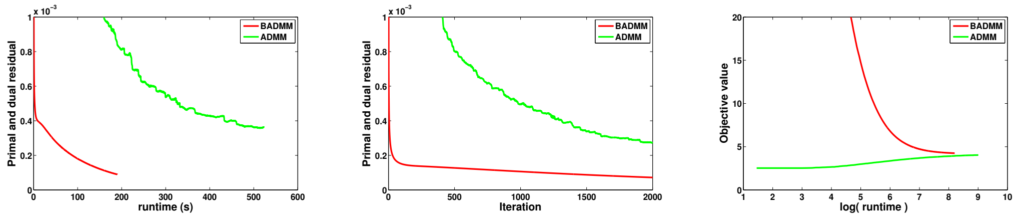
$$\mathbf{Z}^{t+1} = \underset{\mathbf{Z} \in \mathbf{\Delta}_{\mathbf{z}}}{\text{argmin}} \langle \mathbf{Y}^t, -\mathbf{Z} \rangle + \rho \text{KL}(\mathbf{Z}, \mathbf{X}^{t+1}) \ , \tag{4.59}$$

$$\mathbf{Y}^{t+1} = \mathbf{Y}^t + \rho(\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}) \ . \tag{4.60}$$

Both (4.58) and (4.59) have closed-form solutions, i.e.,

$$X_{ij}^{t+1} = \frac{Z_{ij}^t \exp(-\frac{C_{ij}+Y_{ij}^t}{\rho})}{\sum_{j=1}^n Z_{ij}^t \exp(-\frac{C_{ij}+Y_{ij}^t}{\rho})} a_i \ , \quad Z_{ij}^{t+1} = \frac{X_{ij}^{t+1} \exp(\frac{Y_{ij}^t}{\rho})}{\sum_{i=1}^m X_{ij}^{t+1} \exp(\frac{Y_{ij}^t}{\rho})} b_j \tag{4.61}$$

which are exponentiated graident updates and can be done in $O(mn)$. Besides the sum operation which can be done in $O(\log(n))$, (4.61) amounts to elementwise operation and thus can be done in parallel. According to Corollary 1, BADMM can be faster than ADMM by a factor of $O(n/\log(n))$.



(a) m = n = 1000 (b) m = n = 2000 (c) m = n = 4000

Figure 4.1: Comparison BADMM and ADMM. BADMM converges faster than ADMM.

We compare BADMM with ADMM and a highly optimized commercial linear programming solvers on the mass transportation problem (4.56) when $m = n$ and $\mathbf{a} = \mathbf{b} = \mathbf{e}$. $\mathbf{C}$ is randomly generated from the uniform distribution. They run 5 times and the average is reported. We choose the 'best' parameter for BADMM ($\rho = 0.001$) and ADMM ($\rho = 0.001$). The stopping condition is either when the number of iterations exceeds 2000 or when the primal-dual residual is less than 10^{-4}.

BADMM vs ADMM: Figure 4.1 compares BADMM and ADMM with different dimensions $n = \{1000, 2000, 4000\}$ running on a single CPU. Figure 4.1(a) plots the primal and dual residual against the runtime when the dimension is 1000, and Figure 4.1(b) plots the convergence of primal and dual residual over iteration when the dimension is 2000. BADMM converges faster than ADMM. Figure 4.1(c) plots the convergence of objective value against the log of runtime. BADMM converges faster than ADMM even when the initial point is further from the optimum.

BADMM vs Gurobi: Gurobi[2] is a highly optimized commercial software where linear programming solvers have been efficiently implemented. We run Gurobi on two settings: a

[2] http://www.gurobi.com/

Table 4.1: Comparison of BADMM (GPU) with Gurobi

m=n	Gurobi (Laptop)		Gurobi (Server)		BADMM (GPU)	
	time	objective	time	objective	time	objective
2^{10}	4.22	1.69	2.66	1.69	0.54	1.69
5×2^{10}	377.14	1.61	92.89	1.61	22.15	1.61
10×2^{10}	-	-	1235.34	1.65	117.75	1.65
15×2^{10}	-	-	-	-	303.54	1.63

Mac laptop with 6G memory and a server with 86G memory, respectively. For comparison, BADMM is run in parallel on a Tesla M2070 GPU with 5G memory and 448 cores[3] . We experiment with large scale problems and use $m = n = \{1, 5, 10, 15\} \times 2^{10}$. Table 1 shows the runtime and the objective values of BADMM and Gurobi, where a '-' indicates the algorithm did not terminate. In spite of Gurobi being one of the most optimized LP solvers, BADMM running in parallel is several times faster than Gurobi. In fact, for larger values of n, Gurobi did not terminate even on the 86G server, whereas BADMM was efficient even with just 5G memory! The complexity of most LP solvers in Gurobi is $O(n^3)$ and can become slow as n increases, especially at the scales we consider. Moreover, the memory consumption of Gurobi increases rapidly with the increase of n. When $n = 5 \times 2^{10}$, the memory required by Gurobi surpassed the memory in the laptop, leading to the rapid increase of time. A similar situation was also observed in the server with 86G when $n = 10 \times 2^{10}$. In contrast, the memory required by BADMM is $O(n^2)$—even when $n = 15 \times 2^{10}$ (more than 0.2 billion parameters), BADMM can still run on a single GPU with only 5G memory.

The results clearly illustrate the promise of BADMM. With more careful implementation and code optimization, BADMM has the potential to solve large scale problems efficiently in parallel with small memory foot-print.

[3] GPU code is available on `http://www-users.cs.umn.edu/~huwang/badmm_mt.zip`

Appexdix

4.A Convergence of BADMM with Time Varying Step Size

Under the assumption that $\mathbf{y}_t$ is bounded, the following theorem requires a large step size to establish the convergence of BADMM.

Theorem 11 *Let the sequences* $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ *be generated by Bregman ADMM (7.21)-(4.8) and* $\{\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*\}$ *satisfying (4.25)-(4.27). Let the Assumption 6 hold and* $\|\mathbf{y}_t\|_2 \leq D_{\mathbf{y}}$. *Setting* $\rho_{\mathbf{x}} = \rho_{\mathbf{z}} = c_1\sqrt{T}, \tau = c_2\sqrt{T}$ *and* $\rho = \sqrt{T}$ *for some positive constant* c_1, c_2, *then* $R(t+1)$ *converges to zero.*

Proof: Assuming $\|\mathbf{y}_t\|_2 \leq D_{\mathbf{y}}$ and using (4.8), we have

$$\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 = \frac{1}{\tau^2}\|\mathbf{y}_{t+1} - \mathbf{y}_t\|_2^2 \leq \frac{2}{\tau^2}(\|\mathbf{y}_{t+1}\|_2^2 + \|\mathbf{y}_t\|_2^2) \leq \frac{4D_{\mathbf{y}}^2}{\tau^2} . \tag{4.62}$$

Plugging into (4.46) and rearranging the terms yields

$$R(t+1) \leq D(\mathbf{w}^*, \mathbf{w}_t) - D(\mathbf{w}^*, \mathbf{w}_{t+1}) + (\frac{\tau}{2\rho} + \gamma)\frac{4D_{\mathbf{y}}^2}{\tau^2} . \tag{4.63}$$

Setting $\rho_{\mathbf{x}} = \rho_{\mathbf{z}} = c_1\sqrt{T}, \tau = c_2\sqrt{T}$ and $\rho = \sqrt{T}$ for some positive constant c_1, c_2, we have

$$\begin{aligned} R(t+1) = &\, c_1 B_{\varphi_{\mathbf{x}}}(\mathbf{x}_{t+1}, \mathbf{x}_t) + c_1 B_{\varphi_{\mathbf{z}}}(\mathbf{z}_{t+1}, \mathbf{z}_t) + B_{\phi}(\mathbf{c} - \mathbf{A}\mathbf{x}_{t+1}, \mathbf{B}\mathbf{z}_t) \\ &+ \gamma\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 , \end{aligned} \tag{4.64}$$

Summing (4.63) over t from 0 to $T-1$, we have the following telescoping sum

$$\sum_{t=0}^{T-1} R(t+1) \leq D(\mathbf{w}^*, \mathbf{w}_0) + \sum_{t=0}^{T-1}(\frac{\tau}{2\rho} + \gamma)\frac{4D_{\mathbf{y}}^2}{\tau^2} = D(\mathbf{w}^*, \mathbf{w}_0) + \frac{4(c_2/2 + \gamma)D_{\mathbf{y}}^2}{c_2^2} . \tag{4.65}$$

Therefore, $R(t+1) \to 0$ as $t \to \infty$. ■

The following theorem establishes the convergence rate for the objective and residual of constraints in an ergodic sense.

Theorem 12 *Let the sequences* $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ *be generated by Bregman ADMM (7.21)-(4.8). Let* $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{x}_t, \bar{\mathbf{z}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{z}_t$. *Let the Assumption 6 hold and* $\|\mathbf{y}_t\|_2 \leq D_{\mathbf{y}}$. *Set* $\rho_{\mathbf{x}} =$

$\rho_{\mathbf{z}} = c_1\sqrt{T}, \tau = c_2\sqrt{T}, \rho = \sqrt{T}$ for some positive constants c_1, c_2. For any $(\mathbf{x}^*, \mathbf{z}^*, \mathbf{y}^*)$ satisfying KKT conditions (4.25)-(4.27), we have

$$f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \frac{2D_{\mathbf{y}}^2}{c_2\sqrt{T}} + \frac{\|\mathbf{y}_0\|_2^2}{2c_2T\sqrt{T}} + \frac{D_2}{\sqrt{T}} , \tag{4.66}$$

$$\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T - \mathbf{c}\|_2^2 \leq \frac{D(\mathbf{w}^*, \mathbf{w}_0)}{\gamma T} + \frac{4(c_2/2+\gamma)D_{\mathbf{y}}^2}{\gamma c_2^2 T} , \tag{4.67}$$

where $D_2 = B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_0) + c_1(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_0) + B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_0))$.

Proof: Assuming $\|\mathbf{y}_t\|_2 \leq D_{\mathbf{y}}^2$ and using (4.8), we have

$$-\langle \mathbf{y}_t, \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\rangle = -\frac{1}{\tau}\langle \mathbf{y}_t, \mathbf{y}_{t+1} - \mathbf{y}_t\rangle \leq \frac{1}{\tau}(\|\mathbf{y}_t\|_2^2 + \|\mathbf{y}_t\|_2 * \|\mathbf{y}_{t+1}\|_2) \leq \frac{2D_{\mathbf{y}}^2}{\tau} . \tag{4.68}$$

Plugging into (4.33) and ignoring some negative terms yield

$$\begin{aligned} & f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \\ & \leq \frac{2D_{\mathbf{y}}^2}{\tau} + \rho(B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_t) - B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_{t+1})) + \rho_{\mathbf{x}}(B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_t) - B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_{t+1})) \\ & + \rho_{\mathbf{z}}(B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_t) - B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_{t+1})) . \end{aligned} \tag{4.69}$$

Summing over t from 0 to $T-1$, we have the following telescoping sum

$$\begin{aligned} & \sum_{t=0}^{T-1} [f(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))] \\ & \leq \sum_{t=0}^{T-1} \frac{2D_{\mathbf{y}}^2}{\tau} + \frac{1}{2\tau}\|\mathbf{y}_0\|_2^2 + \rho B_\phi(\mathbf{B}\mathbf{z}^*, \mathbf{B}\mathbf{z}_0) + \rho_{\mathbf{x}} B_{\varphi_{\mathbf{x}}}(\mathbf{x}^*, \mathbf{x}_0) + \rho_{\mathbf{z}} B_{\varphi_{\mathbf{z}}}(\mathbf{z}^*, \mathbf{z}_0) . \end{aligned}$$

Setting $\rho_{\mathbf{x}} = \rho_{\mathbf{z}} = c_1\sqrt{T}, \tau = c_2\sqrt{T}, \rho = \sqrt{T}$, dividing both sides by T and applying the Jensen's inequality yield (4.66).

Dividing both sides of (4.65) by T and applying the Jesen's inequality yield (4.67). ■

Chapter 5

Parallel Direction Method of Multipliers

5.1 Introduction

In this chapter, we consider the minimization of block-seperable convex functions subject to linear constraints, with a canonical form:

$$\min_{\{\mathbf{x}_j \in \mathcal{X}_j\}} f(\mathbf{x}) = \sum_{j=1}^{J} f_j(\mathbf{x}_j) \ , \ \text{s.t. } \mathbf{A}\mathbf{x} = \sum_{j=1}^{J} \mathbf{A}_j^c \mathbf{x}_j = \mathbf{a} \ , \tag{5.1}$$

where the objective function $f(\mathbf{x})$ is a sum of J block separable (nonsmooth) convex functions, $\mathbf{A}_j^c \in \mathbb{R}^{m \times n_j}$ is the j-th column block of $\mathbf{A} \in \mathbb{R}^{m \times n}$ where $n = \sum_j n_j$, $\mathbf{x}_j \in \mathbb{R}^{n_j \times 1}$ is the j-th block coordinate of $\mathbf{x}$, $\mathcal{X}_j$ is a local convex constraint of $\mathbf{x}_j$ and $\mathbf{a} \in \mathbb{R}^{m \times 1}$. The canonical form can be extended to handle linear inequalities by introducing slack variables, i.e., writing $\mathbf{A}\mathbf{x} \leq \mathbf{a}$ as $\mathbf{A}\mathbf{x} + \mathbf{z} = \mathbf{a}, \mathbf{z} \geq \mathbf{0}$.

A variety of machine learning problems can be cast into the linearly-constrained optimization problem (5.1). For example, in robust Principal Component Analysis (RPCA) [24], one attempts to recover a low rank matrix $\mathbf{L}$ and a sparse matrix $\mathbf{S}$ from an observation matrix M, i.e., the linear constraint is $\mathbf{M} = \mathbf{L} + \mathbf{S}$. Further, in the stable version of RPCA [223], an noisy matrix $\mathbf{Z}$ is taken into consideration, and the linear constraint has three blocks, i.e., $\mathbf{M} = \mathbf{L} + \mathbf{S} + \mathbf{Z}$. The linear constraint with three blocks also appears in the latent variable

Gaussian graphical model selection problem [29, 127]. Problem (5.1) can also include composite minimization problems which solve a sum of a loss function and a set of nonsmooth regularization functions. Due to the increasing interest in structural sparsity [4], composite regularizers have become widely used, e.g., overlapping group lasso [222]. As the blocks are overlapping in this class of problems, it is difficult to apply block coordinate descent methods for large scale problem [148, 160] which assume block-separable. By simply splitting blocks through introducing equality constraints, the composite minimization problem can also formulated as (5.1) [19].

A classical approach to solving (5.1) is to relax the linear constraints using the (augmented) Lagrangian [163, 164], i.e.,

$$L_\rho(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} - \mathbf{a} \rangle + \frac{\rho}{2} \|\mathbf{A}\mathbf{x} - \mathbf{a}\|_2^2 \,. \tag{5.2}$$

where $\rho \geq 0$ is called the penalty parameter. We call $\mathbf{x}$ the primal variable and $\mathbf{y}$ the dual variable. (5.2) usually leads to primal-dual algorithms which update the primal and dual variables alternatively. The dual update is simply dual gradient ascent where the dual gradient is the resiudal of equality constraint, i.e., $\mathbf{A}\mathbf{x} - \mathbf{a}$. The primal update is to solve a minimization problem of (5.2) given $\mathbf{y}$. The primal update determines the efficiency of this class of primal-dual algorithms and will be the focus of this chapter.

If $\rho = 0$, (5.2) decomposes into J independent subproblems provided f is separable. In this scenario, the primal-dual algorithm is called the dual ascent method [20, 178], where the primal update is solved in a parallel block coordinate fashion. While the dual ascent method can achieve massive parallelism, a careful choice of stepsize and some strict conditions are required for convergence, particularly when f is nonsmooth. To achieve better numerical efficiency and convergence behavior compared to the dual ascent method, it is favorable to set $\rho > 0$ in the augmented Lagrangian (5.2). However, (5.2) is no longer separable since the augmentation term makes $\mathbf{x}$ coupled. A well-known primal-dual algorithm to solve (5.2) is the method of multipliers, which solves the primal update in one block. For large scale optimization problems, it is often difficult to solve the entire augmented Lagrangian efficiently. Considerable efforts have thus been devoted to solving the primal update of the method of multipliers efficiently. In [186], randomized block coordinate descent (RBCD) [148, 160] is used to solve (5.2) exactly, but leading to a double-loop algorithm along with the dual step. More recent results show (5.2) can be solved inexactly by just sweeping the coordinates once using the alternating direction method of multipliers (ADMM) [67, 19].

When $J = 2$, the constraint is of the form $\mathbf{A}_1^c \mathbf{x}_1 + \mathbf{A}_2^c \mathbf{x}_2 = \mathbf{a}$. In this case, it has been shown that ADMM can solve the augmented Lagrangian seperately and alternatively. Encouraged by the success of ADMM with two blocks [19], ADMM has also been extended to solve the problem with multiple blocks [86, 85, 47, 153, 78, 32]. The variants of ADMM can be mainly divided into two categories. One is Gauss-Seidel ADMM (GSADMM) [86, 85], which solves (5.2) in a cyclic block coordinate manner. [86] established a linear convergence rate for MADMM under some fairly strict conditions: (1) $\mathbf{A}_j$ has full column rank; (2) f_j has Lipschitz-continuous gradients; (3) certain local error bounds hold; (4) the step size needs to be sufficiently small. In [78], a back substitution step was added so that the convergence of ADMM for multiple blocks can be proved. In some cases, it has been shown that ADMM might not converge for multiple blocks [32]. In [85], a block successive upper bound minimization method of multipliers (BSUMM) is proposed to solve the problem (5.1). The convergence of BSUMM is established under conditions: (i) certain local error bounds hold; (ii) the step size is either sufficiently small or decreasing. However, in general, Gauss-Seidel ADMM with multiple blocks is not well understood and its iteration complexity is largely open. The other is Jacobi ADMM [207, 47, 153], which solves (5.2) in a parallel block coordinate fashion. In [207, 153], (5.1) is solved by using two-block ADMM with splitting variables (sADMM). [47] considers a proximal Jacobian ADMM (PJADMM) by adding proximal terms. In addition to the two types of extensions, a randomized block coordinate variant of ADMM named RBSUMM was proposed in [85]. However, RBSUMM can only randomly update one block. Moreover, the convergence of RBSUMM is established under the same conditions as BSUMM and its iteration complexity is unknown. In [182], ADMM with stochastic dual coordinate ascent is proposed to solve online or stochastic ADMM [200, 152, 183] problem in the dual, which is not the focus of this chapter.

In this chapter, we propose a randomized block coordinate method named parallel direction method of multipliers (PDMM) which randomly picks up any number of blocks to update in parallel, behaving like randomized block coordinate descent [148, 160]. Like the dual ascent method, PDMM solves the primal update in a parallel block coordinate fashion even with the augmentation term. Moreover, PDMM inherits the merits of the method of multipliers and can solve a fairly large class of problems, including nonsmooth functions. Technically, PDMM has three aspects which make it distinct from such state-of-the-art methods. First, if block coordinates of the primal $\mathbf{x}$ is solved exactly, PDMM uses a backward step on the dual update so that the dual variable makes conservative progress. Second, the sparsity of $\mathbf{A}$ and the number of primal blocks and dual blocks to be updated are taken into consideration to determine the step size of the dual update. Third, PDMM can randomly choose arbitrary number of primal blocks and dual blocks for update in parallel. Moreover, we show that sADMM and PJADMM are the two extreme cases of PDMM. The connection between sADMM and PJADMM through PDMM provides better understanding of dual backward step. PDMM can also be used to solve overlapping groups in a randomized block coordinate fashion. Interestingly, the corresponding problem for RBCD [148, 160] with overlapping blocks is still an open problem. We establish the global convergence and $O(1/T)$ iteration complexity of PDMM with constant step size. We evaluate the performance of PDMM in two applications: robust principal component analysis and overlapping group lasso.

The rest of this chapter is organized as follows. PDMM is proposed in Section 5.2. The convergence results are established in Section 5.3. We evaluate the performance of PDMM in Section 5.4. The proof of the convergence of PDMM is given in the Appendix.

Notations: Assume that $\mathbf{A} \in \mathbb{R}^{m \times n}$ is divided into $I \times J$ blocks. Let $\mathbf{A}_i^r \in \mathbb{R}^{m_i \times n}$ be the i-th row block of $\mathbf{A}$, $\mathbf{A}_j^c \in \mathbb{R}^{m \times n_j}$ be the j-th column block of $\mathbf{A}$, and $\mathbf{A}_{ij} \in \mathbb{R}^{m_i \times n_j}$ be the ij-th block of $\mathbf{A}$. Let $\mathbf{y}_i \in \mathbb{R}^{m_i \times 1}$ be the i-th block coordinate of $\mathbf{y} \in \mathbb{R}^{m \times 1}$. $\mathcal{N}(i)$ is a set of nonzero blocks $\mathbf{A}_{ij}$ in the i-th row block $\mathbf{A}_i^r$ and $d_i = |\mathcal{N}(i)|$ is the number of nonzero blocks. $\lambda_{\max}^{ij}$ is the largest eigenvalue of $\mathbf{A}_{ij}^T \mathbf{A}_{ij}$. $\text{diag}(\mathbf{x})$ denotes a diagonal matrix of vector $\mathbf{x}$. $\mathbf{I}_n$ is an identity matrix of size $n \times n$. Let $\tilde{K}_i = \min\{d_i, K\}$ where K is the number of blocks randomly chosen by PDMM and T be the number of iterations.

5.2 Parallel Direction Method of Multipliers

Consider a direct Jacobi version of ADMM which updates all blocks in parallel:

$$\mathbf{x}_j^{t+1} = \underset{\mathbf{x}_j \in \mathcal{X}_j}{\text{argmin}}\ L_\rho(\mathbf{x}_j, \mathbf{x}_{k \neq j}^t, \mathbf{y}^t)\ , \tag{5.3}$$

$$\mathbf{y}^{t+1} = \mathbf{y}^t + \tau\rho(\mathbf{A}\mathbf{x}^{t+1} - \mathbf{a})\ . \tag{5.4}$$

where τ is a shrinkage factor for the step size of the dual gradient ascent update. However, empirical results show that it is almost impossible to make the direct Jacobi updates (5.3)-(5.4)

Table 5.1: Parameters (τ_i, ν_i) of PDMM. K is the number of primal blocks randomly chosen from J primal blocks, K_I is the number of dual blocks randomly chosen from I dual blocks. $\tilde{K}_i = \min\{d_i, K\}$ where d_i is the number of nonzero blocks $\mathbf{A}_{ij}$ in the i-th row of $\mathbf{A}$.

K	ν_i	τ_i
$K = 1$	0	$\frac{I}{(2J-1)K_I+I-K_I}$
$1 < K < J$	$1 - \frac{1}{\tilde{K}_i}$	$\frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I}+K(1-\frac{K_I}{I})]}$
$K = K_I, I = J$	$1 - \frac{1}{\tilde{K}_i}$	$\frac{J}{\tilde{K}_i(3J-2K)}$
$K = J$	$1 - \frac{1}{d_i}$	$\frac{1}{d_i}$

to converge even when τ is extremely small. [86, 47] also noticed that the direct Jacobi updates may not converge.

To address the problem in (5.3) and (5.4), we propose a backward step on the dual update. Moreover, instead of updating all blocks, the blocks $\mathbf{x}_j$ will be updated in a parallel randomized block coordinate fashion. We call the algorithm Parallel Direction Method of Multipliers (PDMM). At time $t+1$, PDMM first randomly select K primal blocks denoted by $\mathbb{J}_t$ and K_I dual blocks denoted by set $\mathbb{I}_t$, then executes the following iterates:

$$\hat{\mathbf{y}}_i^t = \mathbf{y}_i^t - \nu_i \rho(\mathbf{A}_i \mathbf{x}^t - \mathbf{a}_i) \;, \tag{5.5}$$

$$\mathbf{x}_{j_t}^{t+1} = \underset{\mathbf{x}_{j_t} \in \mathcal{X}_{j_t}}{\operatorname{argmin}} \; L_\rho(\mathbf{x}_{j_t}, \mathbf{x}_{k \neq j_t}^t, \hat{\mathbf{y}}^t) + \eta_{j_t} B_{\phi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) \;,\; j_t \in \mathbb{J}_t \;, \tag{5.6}$$

$$\mathbf{y}_{i_t}^{t+1} = \mathbf{y}_{i_t}^t + \tau_{i_t} \rho(\mathbf{A}_{i_t} \mathbf{x}^{t+1} - \mathbf{a}_{i_t}) \;,\; i_t \in \mathbb{I}_t \;, \tag{5.7}$$

where $\tau_i > 0, 0 \leq \nu_i < 1, \eta_{j_t} \geq 0$, and $B_{\phi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t)$ is a Bregman divergence. Note $\mathbf{x} = [\mathbf{x}_{j_t \in \mathbb{J}_t}^{t+1}, \mathbf{x}_{k \notin \mathbb{J}_t}]^T$ and $\mathbf{y} = [\mathbf{y}_{i_t \in \mathbb{I}_t}^{t+1}, \mathbf{y}_{k \notin \mathbb{I}_t}]^T$. Table 5.1 shows how to choose τ_i and ν_i under different numbers of random primal blocks K, random dual blocks K_I, and block sparsity of $\mathbf{A}$. K is the number of blocks randomly chosen from J blocks, and $\tilde{K}_i = \min\{d_i, K\}$ where d_i is the number of nonzero blocks $\mathbf{A}_{ij}$ in the i-th row of $\mathbf{A}$.

In the $\mathbf{x}_{j_t}$-update (5.6), a Bregman divergence is addded so that exact PDMM and its inexact variants can be analyzed in an unified framework [201]. In particular, if $\eta_{j_t} = 0$, (5.6) is an exact update. If $\eta_{j_t} > 0$, by choosing a suitable Bregman divergence, (5.6) can be solved by various inexact updates, often yielding a closed-form for the $\mathbf{x}_{j_t}$ update (see Section 5.2.1).

Algorithm 4 Parallel Diretion Method of Multipliers

```
1: Input: $\rho, \eta_j, \tau_i, \nu_i$
2: Initialization: $\mathbf{x}^1, \hat{\mathbf{y}}^1 = \mathbf{0}$
3: if $\tau_i, \nu_i$ are not defined, initialize $\tau_i, \nu_i$ as given in Table 5.1
4: $\mathbf{r}^1 = \mathbf{A}\mathbf{x}^1 - \mathbf{a} = -\mathbf{a}$
5: for $t = 1$ to $T$ do
6:     randomly pick up $j_t$ and $i_t$ block coordinates
7:     $\hat{\mathbf{y}}_i^t = \mathbf{y}_i^t - \nu_i \rho \mathbf{r}_i^t$
8:     $\mathbf{x}_{j_t}^{t+1} = \underset{\mathbf{x}_{j_t} \in \mathcal{X}_{j_t}}{\operatorname{argmin}} f_{j_t}(\mathbf{x}_{j_t}) + \langle (\mathbf{A}_{j_t}^c)^T(\hat{\mathbf{y}}^t + \rho \mathbf{r}^t), \mathbf{x}_{j_t} \rangle + \frac{\rho}{2}\|\mathbf{A}_{j_t}^c(\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t)\|_2^2 + \eta_{j_t} B_{\phi_{j_t}}(\mathbf{x}, \mathbf{x}_{j_t}^t)$
9:     $\mathbf{r}^{t+1} = \mathbf{r}^t + \sum_{j_t \in \mathbb{J}_t} \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)$
10:    $\mathbf{y}_{i_t}^{t+1} = \mathbf{y}_{i_t}^t + \tau_{i_t} \rho \mathbf{r}_{i_t}^{t+1}, \ i_t \in \mathbb{I}_t$
11: end for
```

Let $\mathbf{r}^t = \mathbf{A}\mathbf{x}^t - \mathbf{a}$, then $\mathbf{r}^{t+1} = \mathbf{r}^t + \sum_{j_t \in \mathbb{J}_t} \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)$. (5.6) can be rewritten as

$$\begin{aligned}\mathbf{x}_{j_t}^{t+1} &= \underset{\mathbf{x}_{j_t} \in \mathcal{X}_{j_t}}{\operatorname{argmin}} f_{j_t}(\mathbf{x}_{j_t}) + \langle \hat{\mathbf{y}}^t, \mathbf{A}_{j_t}^c \mathbf{x}_{j_t} \rangle + \frac{\rho}{2}\|\mathbf{A}_{j_t}^c \mathbf{x}_{j_t} + \sum_{k \neq j_t} \mathbf{A}_k^c \mathbf{x}_k^t - \mathbf{a}\|_2^2 + \eta_{j_t} B_{\phi_{j_t}}(\mathbf{x}, \mathbf{x}_{j_t}^t) \\ &= \underset{\mathbf{x}_{j_t} \in \mathcal{X}_{j_t}}{\operatorname{argmin}} f_{j_t}(\mathbf{x}_{j_t}) + \langle (\mathbf{A}_{j_t}^c)^T(\hat{\mathbf{y}}^t + \rho \mathbf{r}^t), \mathbf{x}_{j_t} \rangle + \frac{\rho}{2}\|\mathbf{A}_{j_t}^c(\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t)\|_2^2 + \eta_{j_t} B_{\phi_{j_t}}(\mathbf{x}, \mathbf{x}_{j_t}^t) \,. \end{aligned} \tag{5.8}$$

Therefore, we have the algorithm of PDMM as in Algorithm 4.

To better understand PDMM, we discuss the following three aspects which play roles in choosing τ_i and ν_i: the dual backward step (5.5), the sparsity of $\mathbf{A}$ and the choice of randomized blocks.

Dual Backward Step: We attribute the failure of the Jacobi updates (5.3)-(5.4) to the following observation in (5.3), which can be rewritten as:

$$\mathbf{x}_j^{t+1} = \underset{\mathbf{x}_j \in \mathcal{X}_j}{\operatorname{argmin}} f_j(\mathbf{x}_j) + \langle \mathbf{y}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_j^c \mathbf{x}_j \rangle + \frac{\rho}{2}\|\mathbf{A}_j^c(\mathbf{x}_j - \mathbf{x}_j^t)\|_2^2 \,. \tag{5.9}$$

In the primal $\mathbf{x}_j$ update, the quadratic penalty term implicitly adds full gradient ascent step to the dual variable, i.e., $\mathbf{y}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a})$, which we call implicit dual ascent. The implicit dual ascent along with the explicit dual ascent (5.4) may lead to too aggressive progress on the dual variable, particularly when the number of blocks is large. Based on this observation, we introduce an intermediate variable $\hat{\mathbf{y}}^t$ to replace $\mathbf{y}^t$ in (5.9) so that the implicit dual ascent in (5.9) makes conservative progress, e.g., $\hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}) = \mathbf{y}^t + (1-\nu)\rho(\mathbf{A}\mathbf{x}^t - \mathbf{a})$, where $0 < \nu < 1$. $\hat{\mathbf{y}}^t$ is the result of a 'backward step' on the dual variable, i.e., $\hat{\mathbf{y}}^t = \mathbf{y}^t - \nu\rho(\mathbf{A}\mathbf{x}^t - \mathbf{a})$.

Moreover, one can show that τ and ν have also been implicitly used when using two-block ADMM with splitting variables (sADMM) to solve (5.1) [153, 207]. Section 5.2.2 shows sADMM is a special case of PDMM. The connection helps in understanding the role of the two parameters τ_i, ν_i in PDMM. Interestingly, the step sizes τ_i and ν_i can be improved by considering the block sparsity of $\mathbf{A}$ and the number of random blocks to be updated.

Sparsity of A: Assume $\mathbf{A}$ is divided into $I \times J$ blocks. While $\mathbf{x}_j$ can be updated in parallel, the matrix multiplication $\mathbf{A}\mathbf{x}$ in the dual update (5.4) requires synchronization to gather messages from all block coordinates $j_t \in \mathbb{J}_t$. For updating the i-th block of the dual $\mathbf{y}_i$, we need $\mathbf{A}_i\mathbf{x}^{t+1} = \sum_{j_t \in \mathbb{J}_t} \mathbf{A}_{ij_t}\mathbf{x}_{j_t}^{t+1} + \sum_{k \notin \mathbb{J}_t} \mathbf{A}_{ik}\mathbf{x}_k^t$ which aggregates "messages" from all $\mathbf{x}_{j_t}$. If $\mathbf{A}_{ij_t}$ is a block of zeros, there is no "message" from $\mathbf{x}_{j_t}$ to $\mathbf{y}_i$. More precisely, $\mathbf{A}_i\mathbf{x}^{t+1} = \sum_{j_t \in \mathbb{J}_t \cap \mathcal{N}(i)} \mathbf{A}_{ij_t}\mathbf{x}_{j_t}^{t+1} + \sum_{k \notin \mathbb{J}_t} \mathbf{A}_{ik}\mathbf{x}_k^t$ where $\mathcal{N}(i)$ denotes a set of nonzero blocks in the i-th row block $\mathbf{A}_i$. $\mathcal{N}(i)$ can be considered as the set of neighbors of the i-th dual block $\mathbf{y}_i$ and $d_i = |\mathcal{N}(i)|$ is the degree of the i-th dual block $\mathbf{y}_i$. If $\mathbf{A}$ is sparse, d_i could be far smaller than J. According to Table 5.1, a low d_i will lead to bigger step sizes τ_i for the dual update and smaller step sizes for the dual backward step (5.5). Further, as shown in Section 5.2.3, when using PDMM with all blocks to solve composite minimization with overlapping blocks, PDMM can use $\tau_i = 0.5$ which is much larger than $1/J$ in sADMM.

Randomized Blocks: The number of blocks to be randomly chosen also has the effect on τ_i, ν_i. If randomly choosing one primal block ($K = 1$), then $\nu_i = 0$ and thus the dual backward step (5.5) vanishes. If further randomly updating one dual block ($K_I = 1$) and assuming $I = J$, $\tau_i > \frac{1}{3}$. In general, for a particular K_I, τ_i increases as K decreases. For a particular K, τ_i increases as K_I decreases. However, if updating all primal blocks ($K = J$), no matter how many dual blocks are updated, $\tau_i = \frac{1}{d_i}, \nu_i = 1 - \frac{1}{d_i}$.

5.2.1 Inexact PDMM

If $\eta_{j_t} > 0$, there is an extra Bregman divergence term in (5.6), which can serve two purposes. First, choosing a suitable Bregman divergence can lead to a closed-form solution for (5.6). Second, if η_{j_t} is sufficiently large, the dual update can use a large step size ($\tau_i = 1$) and the backward step (5.5) can be removed ($\nu_i = 0$), leading to the same updates as PJADMM [47] (see Section 5.2.2).

Given a differentiable function ψ_{j_t}, its Bregman divergence is defiend as

$$B_{\psi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) = \psi_{j_t}(\mathbf{x}_{j_t}) - \psi_{j_t}(\mathbf{x}_{j_t}^t) - \langle \nabla \psi_{j_t}(\mathbf{x}_{j_t}^t), \mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t \rangle, \tag{5.10}$$

where $\nabla \psi_{j_t}$ denotes the gradient of ψ_{j_t}. Rearranging the terms yields

$$\psi_{j_t}(\mathbf{x}_{j_t}) - B_{\psi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) = \psi_{j_t}(\mathbf{x}_{j_t}^t) + \langle \nabla \psi_{j_t}(\mathbf{x}_{j_t}^t), \mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t \rangle, \tag{5.11}$$

which is exactly the linearization of $\psi_{j_t}(\mathbf{x}_{j_t})$ at $\mathbf{x}_{j_t}^t$. Therefore, if solving (5.6) exactly becomes difficult due to some problematic terms, we can use the Bregman divergence to linearize these problematic terms so that (5.6) can be solved efficiently. More specifically, in (5.6), we can choose $\phi_{j_t} = \varphi_{j_t} - \frac{1}{\eta_{j_t}} \psi_{j_t}$ assuming ψ_{j_t} is the problematic term. Using the linearity of Bregman divergence,

$$B_{\phi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) = B_{\varphi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) - \frac{1}{\eta_{j_t}} B_{\psi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) \ . \tag{5.12}$$

For instance, if f_{j_t} is a logistic function, solving (5.6) exactly requires an iterative algorithm. Setting $\psi_{j_t} = f_{j_t}$, $\varphi_{j_t} = \frac{1}{2}\|\cdot\|_2^2$ in (5.12) and plugging into (5.6) yield

$$\begin{aligned} \mathbf{x}_{j_t}^{t+1} = \underset{\mathbf{x}_{j_t} \in \mathcal{X}_{j_t}}{\operatorname{argmin}} & \ \langle \nabla f_{j_t}(\mathbf{x}_{j_t}^t), \mathbf{x}_{j_t} \rangle + \langle \hat{\mathbf{y}}^t, \mathbf{A}_{j_t} \mathbf{x}_{j_t} \rangle \\ & + \frac{\rho}{2} \|\mathbf{A}_{j_t} \mathbf{x}_{j_t} + \sum_{k \neq j} \mathbf{A}_k \mathbf{x}_k^t - \mathbf{a}\|_2^2 + \eta_{j_t} \|\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t\|_2^2 \ , \end{aligned} \tag{5.13}$$

which has a closed-form solution. Similarly, if the quadratic penalty term $\frac{\rho}{2}\|\mathbf{A}_{j_t}^c \mathbf{x}_{j_t} + \sum_{k \neq j} \mathbf{A}_k^c \mathbf{x}_{j_t} - \mathbf{a}\|_2^2$ is a problematic term, we can set $\psi_{j_t}(\mathbf{x}_{j_t}) = \frac{\rho}{2}\|\mathbf{A}_{j_t}^c \mathbf{x}_{j_t}\|_2^2$, then $B_{\psi_{j_t}}(\mathbf{x}_{j_t}, \mathbf{x}_{j_t}^t) = \frac{\rho}{2}\|\mathbf{A}_{j_t}^c(\mathbf{x}_{j_t} - \mathbf{x}_{j_t}^t)\|_2^2$ can be used to linearize the quadratic penalty term.

In (5.12), the nonnegativeness of $B_{\phi_{j_t}}$ implies that $B_{\varphi_{j_t}} \geq \frac{1}{\eta_{j_t}} B_{\psi_{j_t}}$. This condition can be satisfied as long as φ_{j_t} is more convex than ψ_{j_t}. Technically, we assume that φ_{j_t} is σ/η_{j_t}-strongly convex and ψ_{j_t} has Lipschitz continuous gradient with constant σ, which has been shown in [201]. For instance, if $\psi_{j_t}(\mathbf{x}_{j_t}) = \frac{\rho}{2}\|\mathbf{A}_{j_t}^c \mathbf{x}_{j_t}\|_2^2$, $\sigma = \rho \lambda_{\max}(\mathbf{A}_{j_t}^c)$ where $\lambda_{\max}(\mathbf{A}_{j_t}^c)$ denotes the largest eigenvalue of $(\mathbf{A}_{j_t}^c)^T \mathbf{A}_{j_t}^c$. If choosing $\varphi_{j_t} = \frac{1}{2}\|\cdot\|_2^2$, the condition is satisfied by setting $\eta_{j_t} \geq \rho \lambda_{\max}(\mathbf{A}_{j_t}^c)$.

5.2.2 Connections to Related Work

All blocks: There are also two other methods which update all blocks in parallel. If solving the primal updates exactly, two-block ADMM with splitting variables (sADMM) is considered in [153, 207]. We show that sADMM is a special case of PDMM when setting $\tau_i = \frac{1}{J}$ and $\nu_i = 1 - \frac{1}{J}$ (See Appendix 5.B). If the primal updates are solved inexactly, [47] considers a proximal Jacobian ADMM (PJADMM) by adding proximal terms where the converge rate is improved to $o(1/T)$ given the sufficiently large proximal terms. We show that PJADMM [47] is also a special case of PDMM (See Appendix 5.C). sADMM and PJADMM are two extreme cases of PDMM. The connection between sADMM and PJADMM through PDMM can provide better understanding of the three methods and the role of dual backward step. If the primal update is solved exactly which makes sufficient progress, the dual update should take small step, e.g., sADMM. On the other hand, if the primal update takes small progress by adding proximal terms, the dual update can take full gradient step, e.g. PJADMM. While sADMM is a direct derivation of ADMM, PJADMM introduces more terms and parameters.

Randomized blocks: While PDMM can randomly update any number of blocks, RBUSMM [85] can only randomly update one block. The convergence of RBSUMM requires certain local error bounds to be hold and decreasing step size. Moreover, the iteration complexity of RBSUMM is still unknown. In contast, PDMM converges at a rate of $O(1/T)$ with the constant step size.

5.2.3 Randomized Overlapping Block Coordinate

Consider the composite minimization problem of a sum of a loss function $\ell(\mathbf{w})$ and composite regularizers $g_j(\mathbf{w}_j)$:

$$\min_{\mathbf{w}} \ \ell(\mathbf{w}) + \sum_{j=1}^{L} g_j(\mathbf{w}_j) \ , \tag{5.14}$$

which considers L overlapping groups $\mathbf{w}_j \in \mathbb{R}^{b\times 1}$. Let $J = L+1, \mathbf{x}_J = \mathbf{w}$. For $1 \leq j \leq L$, denote $\mathbf{x}_j = \mathbf{w}_j$, then $\mathbf{x}_j = \mathbf{U}_j^T \mathbf{x}_J$, where $\mathbf{U}_j \in \mathbb{R}^{b\times L}$ is the columns of an identity matrix and extracts the coordinates of $\mathbf{x}_J$. By letting $f_j(\mathbf{x}_j) = g_j(\mathbf{w}_j)$ and $f_J(\mathbf{x}_J) = \ell(\mathbf{w})$, (5.14) can be written as:

$$\min_{\mathbf{x}} \ \sum_{j=1}^{J} f_j(\mathbf{x}_j) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{I} & & & -\mathbf{U}_1 \\ & \ddots & & \vdots \\ & & \mathbf{I} & -\mathbf{U}_L \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_L \\ \mathbf{x}_J \end{bmatrix} = \mathbf{0}. \tag{5.15}$$

where $\mathbf{x} = [\mathbf{x}_1; \cdots ; \mathbf{x}_L; \mathbf{x}_{L+1}] \in \mathbb{R}^{b \times J}$. $\mathbf{x}_J$ is a global variable and $\mathbf{x}_j, 1 \leq j \leq L$ is a local variable. For a local variable $\mathbf{x}_j$, the step 8 in Algorithm 4 can be reduced to

$$\mathbf{x}_j^{t+1} = \underset{\mathbf{x}_j \in \mathcal{X}_j}{\operatorname{argmin}} f_j(\mathbf{x}_j) + \langle \hat{\mathbf{y}}_j^t + \rho \mathbf{r}_j^t, \mathbf{x}_j \rangle + \frac{\rho}{2} \|\mathbf{x}_j - \mathbf{x}_j^t\|_2^2 + \eta_j B_{\phi_j}(\mathbf{x}, \mathbf{x}_j^t) \ . \tag{5.16}$$

where $\hat{\mathbf{y}}_j^t = \mathbf{y}_j^t - \nu_j \mathbf{r}_j^t$ and $\mathbf{r}_j^t = \mathbf{x}_j^t - \mathbf{U}_j \mathbf{x}_J^t$. Assume $i_t = j_t, \mathbf{y}_{j_t}^t = \mathbf{y}_{j_t}^{t-1} + \tau_{j_t}(\mathbf{x}_{j_t}^{t-1} - \mathbf{U}_{j_t} \mathbf{x}_J^{t-1})$. Otherwise, the dual variable $\mathbf{y}_{j_t}$ is not going to be updated. Therefore, if the global variable $\mathbf{x}_J$ is not picked in the history, PDMM does not require the synchronization in the updates of local variables. In contrast, ADMM requires synchronization at each iteration [19]. If the global variable is selected, its update requires the aggregation of newest information of local variables. After the update of global variable, PDMM broadcasts the global information to local variables. Note the aggregation and broadcast steps can be done asynchronously. At time t, only some local variables are updated, aggregation step at time $t + 1$ only acquires those local variables, without the need of synchronization of all local variables. At time $t + 2$, PDMM can first send the global variable to local variables to be selected, without the need of broadcast to all local variables. In summary, PDMM can solve the consensus optimization asynchronously, without the need of synchronization of all variables.

In $\mathbf{A}$, $K_I = K, I = J$. For a row block, there are only two nonzero blocks, i.e., $d_i = 2$. Therefore, $\tau_i = \frac{J}{2(3J-2K)} > \frac{1}{6}, \nu_i = 0.5$. In particular, if $K = J$, $\tau_i = \nu_i = 0.5$. In contrast, sADMM uses $\tau_i = 1/J \ll 0.5, \nu_i = 1 - 1/J > 0.5$ if J is larger.

Remark 3 (a) ADMM [19] can solve (5.15) where the equality constraint is $\mathbf{x}_j = \mathbf{U}_j^T \mathbf{x}_J$.

(b) In this setting, Gauss-Seidel ADMM (GSADMM) and BSUMM [85] are the same as ADMM. BSUMM should converge with constant stepsize ρ (not necessarily sufficiently small), although the theory of BSUMM does not include this special case.

(c) Consensus optimization [19] has the same formulation as (5.15). Therefore, PDMM can also be used as a randomized consensus optimization algorithm.

5.3 Theoretical Results

We establish the convergence results for PDMM under fairly simple assumptions:

Assumption 7

(1) $f_j : \mathbb{R}^{n_j} \to \mathbb{R} \cup \{+\infty\}$ are closed, proper, and convex.

(2) A KKT point of the Lagrangian ($\rho = 0$ in (5.2)) of Problem (5.1) exists.

Assumption 8 is the same as that required by ADMM [19, 200]. Assume that $\{\mathbf{x}_j^*, \mathbf{y}_i^*\}$ satisfies the KKT conditions of the Lagrangian ($\rho = 0$ in (5.2)), i.e.,

$$-\mathbf{A}_j^T \mathbf{y}^* \in \partial f_j(\mathbf{x}_j^*) \ , \tag{5.17}$$

$$\mathbf{A}\mathbf{x}^* - \mathbf{a} = 0. \tag{5.18}$$

During iterations, (5.83) is satisfied if $\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a}$. Let ∂f_j be the subdifferential of f_j. The optimality conditions for the $\mathbf{x}_j$ update (5.6) is

$$-\mathbf{A}_j^c[\mathbf{y}^t + (1-\nu)\rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}) + \mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)] - \eta_j(\nabla\phi_j(\mathbf{x}_j^{t+1}) - \nabla\phi_j(\mathbf{x}_j^t)) \in \partial f_j(\mathbf{x}_j^{t+1}) \ . \tag{5.19}$$

When $\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a}$, $\mathbf{y}^{t+1} = \mathbf{y}^t$. If $\mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t) = 0$, then $\mathbf{A}\mathbf{x}^t - \mathbf{a} = 0$. When $\eta_j \geq 0$, further assuming $B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) = 0$, (5.82) will be satisfied. Overall, the KKT conditions (5.82)-(5.83) are satisfied if the following optimality conditions are satisfied by the iterates:

$$\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a} \ , \mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t) = 0 \ , \tag{5.20}$$

$$B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) = 0 \ . \tag{5.21}$$

The above optimality conditions are sufficient for the KKT conditions. (5.86) are the optimality conditions for the exact PDMM. (5.87) is needed only when $\eta_j > 0$.

Let $\mathbf{z}_{ij} = \mathbf{A}_{ij}\mathbf{x}_j \in \mathbb{R}^{m_i \times 1}$, $\mathbf{z}_i^r = [\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{iJ}^T]^T \in \mathbb{R}^{m_i J \times 1}$ and $\mathbf{z} = [(\mathbf{z}_1^r)^T, \cdots, (\mathbf{z}_I^r)^T]^T \in \mathbb{R}^{Jm \times 1}$. Define the residual of optimality conditions (5.86)-(5.87) as

$$R(\mathbf{x}^{t+1}) = \frac{\rho}{2}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2 + \frac{\rho}{2}\sum_{i=1}^{I}\beta_i\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 + \sum_{j=1}^{J}\eta_j B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) \ . \tag{5.22}$$

where $\mathbf{P}_t$ is some positive semi-definite matrix[1] and $\beta_i = \frac{K}{J\tilde{K}_i}$. If $R(\mathbf{x}^{t+1}) \to 0$, (5.86)-(5.87) will be satisfied and thus PDMM converges to the KKT point $\{\mathbf{x}^*, \mathbf{y}^*\}$. Define the current iterate $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ and $h(\mathbf{v}^*, \mathbf{v}^t)$ as a distance from $\mathbf{v}^t$ to a KKT point $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$:

$$h(\mathbf{v}^*, \mathbf{v}^t) = \frac{K}{J}\sum_{i=1}^{I}\frac{1}{2\tau_i\rho}\|\mathbf{y}_i^* - \mathbf{y}_i^{t-1}\|_2^2 + \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 + \sum_{j=1}^{J}\eta_j B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) \ , \tag{5.23}$$

[1] See the definition in the Appendix 5.A.

where $\mathbf{Q}$ is a positive semi-definite matrix[1] and $\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t)$ with $\gamma_i = \frac{2(J-K)}{\tilde{K}_i(2J-K)} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i}$ is

$$\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) = f(\mathbf{x}^t) - f(\mathbf{x}^*) + \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle + \frac{(\gamma_i - \tau_i)\rho}{2} \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right\} . \quad (5.24)$$

The following Lemma shows that $h(\mathbf{v}^*, \mathbf{v}^t) \geq 0$.

Lemma 13 *Let $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ be generated by PDMM (5.6)-(5.5) and $h(\mathbf{v}^*, \mathbf{v}^t)$ be defined in (5.23). Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i(2J-K)}$, we have*

$$h(\mathbf{v}^*, \mathbf{v}^t) \geq \frac{\rho}{2} \sum_{i=1}^{I} \zeta_i \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 + \frac{\rho}{2} \|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 + + \sum_{j=1}^{J} \eta_j B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) \geq 0 . \quad (5.25)$$

where $\zeta_i = \frac{J-K}{\tilde{K}_i(2J-K)} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i} \geq 0$. Moreover, if $h(\mathbf{v}^*, \mathbf{v}^t) = 0$, then $\mathbf{A}_i^r \mathbf{x}^t = \mathbf{a}_i, \mathbf{z}^t = \mathbf{z}^*$ and $B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) = 0$. Thus, (5.17)-(5.18) are satisfied.

In PDMM, $\mathbf{y}^{t+1}$ depends on $\mathbf{x}^{t+1}$, which in turn depends on $\mathbb{I}_t$. $\mathbf{x}^t$ and $\mathbf{y}^t$ are independent of $\mathbb{I}_t$. $\mathbf{x}^t$ depends on the observed realizations of the random variable

$$\xi_{t-1} = \{\mathbb{I}_1, \cdots, \mathbb{I}_{t-1}\} . \quad (5.26)$$

The following theorem shows that $h(\mathbf{v}^*, \mathbf{v}^t)$ decreases monotonically and thus establishes the global convergence of PDMM.

Theorem 13 ***(Global Convergence of PDMM)*** *Let $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ be generated by PDMM (5.6)-(5.5) and $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$ be a KKT point satisfying (5.17)-(5.18). Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i(2J-K)}$, we have*

$$0 \leq \mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) \leq \mathbb{E}_{\xi_{t-1}} h(\mathbf{v}^*, \mathbf{v}^t) , \quad \mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \to 0 . \quad (5.27)$$

The following theorem establishes the iteration complexity of PDMM in an ergodic sense.

Theorem 14 *Let $(\mathbf{x}_j^t, \mathbf{y}_i^t)$ be generated by PDMM (5.6)-(5.5). Let $\bar{\mathbf{x}}^T = \sum_{t=1}^{T} \mathbf{x}^t$. Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i(2J-K)}$, we have*

$$\mathbb{E} f(\bar{\mathbf{x}}^T) - f(\mathbf{x}^*) \leq \frac{\frac{J}{K}\left\{ \sum_{i=1}^{I} \frac{1}{2\beta_i\rho} \|\mathbf{y}_i^*\|_2^2 + \tilde{\mathcal{L}}_\rho(\mathbf{x}^1, \mathbf{y}^1) + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^1\|_{\mathbf{Q}}^2 + \sum_{j=1}^{J} \eta_j B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^1) \right\}}{T} , \quad (5.28)$$

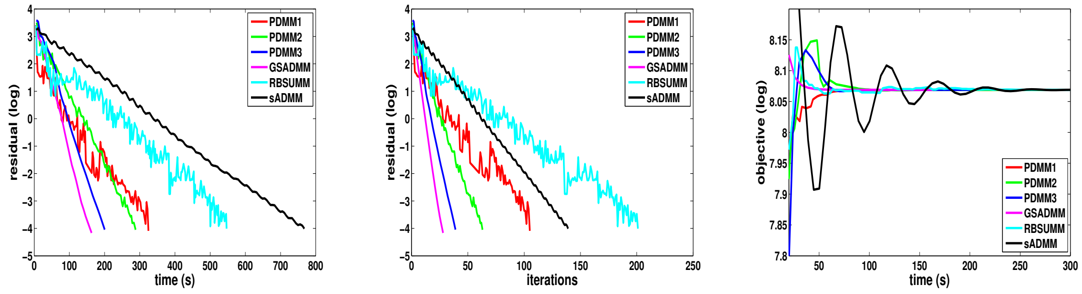


Figure 5.1: Comparison of the convergence of PDMM (with K blocks) with ADMM methods in RPCA. The values of τ_i, ν_i in PDMM is computed according to Table 5.1. Gauss-Seidel (GSADMM) is the fastest algorithm, although whether it converges or not is unknown. PDMM3 is faster than PDMM1 and PDMM2. For the two randomized one block coordinate methods, PDMM1 is faster than RBSUMM.

$$\mathbb{E}\sum_{i=1}^{I}\beta_i\|\mathbf{A}_i^r\bar{\mathbf{x}}^T-\mathbf{a}_i\|_2^2\leq\frac{\frac{2}{\rho}h(\mathbf{v}^*,\mathbf{v}^0)}{T}\ . \tag{5.29}$$

where $\beta_i=\frac{K}{J\tilde{K}_i}$ *and* $\mathbf{Q}$ *is a positive semi-definite matrix.*

5.4 Experimental Results

In this section, we evaluate the performance of PDMM in solving robust principal component analysis (RPCA) and overlapping group lasso [222]. We compared PDMM with ADMM [19] or GSADMM (no theory guarantee), sADMM [153, 207], and RBSUMM [85]. Note GSADMM includes BSUMM [85]. All experiments are implemented in Matlab and run sequentially. We run the experiments 10 times and report the average results. The stopping criterion is either residual $\frac{\text{norm(x-xold)}}{\text{norm(xold)}}+\frac{\text{norm(y-yold)}}{\text{norm(yold)}}\leq 10^{-4}$ or the maximum number of iterations.

RPCA: RPCA is used to obtain a low rank and sparse decomposition of a given matrix $\mathbf{A}$ corrupted by noise [24, 153]:

$$\min\ \frac{1}{2}\|\mathbf{X}_1\|_F^2+\gamma_2\|\mathbf{X}_2\|_1+\gamma_3\|\mathbf{X}_3\|_*\ s.t.\ \mathbf{A}=\mathbf{X}_1+\mathbf{X}_2+\mathbf{X}_3\ . \tag{5.30}$$

where $\mathbf{A}\in\mathbb{R}^{m\times n}$, $\mathbf{X}_1$ is a noise matrix, $\mathbf{X}_2$ is a sparse matrix and $\mathbf{X}_3$ is a low rank matrix.

Table 5.2: The 'best' results of PDMM with tuning parameters τ_i, ν_i in RPCA. PDMM1 randomly updates one block and is the fastest algorithm. PDMMs converges faster than other ADMM methods.

	time (s)	iteration	residual($\times 10^{-5}$)	objective (log)
PDMM1	118.83	40	3.60	8.07
PDMM2	137.46	34	5.51	8.07
PDMM3	147.82	31	6.54	8.07
GSADMM	163.09	28	6.84	8.07
RBSUMM	206.96	141	8.55	8.07
sADMM[2]	731.51	139	9.73	8.07

$\mathbf{A} = \mathbf{L} + \mathbf{S} + \mathbf{V}$ is generated in the same way as [153][2] . In this experiment, $m = 1000, n = 5000$ and the rank is 100. The number appended to PDMM denotes the number of blocks (K) to be chosen in PDMM, e.g., PDMM1 randomly updates one block.

Figure 5.1 compares the convegence results of PDMM with ADMM methods. In PDMM, $\rho = 1$ and τ_i, ν_i are chosen according to Table (5.1), i.e., $(\tau_i, \nu_i) = \{(\frac{1}{5}, 0), (\frac{1}{4}, \frac{1}{2}), (\frac{1}{3}, \frac{1}{3})\}$ for PDMM1, PDMM2 and PDMM3 respectively. We choose the 'best' results for GSADMM ($\rho = 1$) and RBSUMM ($\rho = 1, \alpha = \rho\frac{11}{\sqrt{t}+10}$) and sADMM ($\rho = 1$). PDMMs perform better than RBSUMM and sADMM. Note the public available code of sADMM[2] does not have dual update, i.e., $\tau_i = 0$. sADMM should be the same as PDMM3 if $\tau_i = \frac{1}{3}$. Since $\tau_i = 0$, sADMM is the slowest algorithm. Without tuning the parameters of PDMM, GSADMM converges faster than PDMM. Note PDMM can run in parallel but GSADMM only runs sequentially. PDMM3 is faster than two randomized version of PDMM since the costs of extra iterations in PDMM1 and PDMM2 have surpassed the savings at each iteration. For the two randomized one block coordinate methods, PDMM1 converges faster than RBSUMM in terms of both the number of iterations and runtime.

The effect of τ_i, ν_i: We tuned the parameter τ_i, ν_i in PDMMs. Three randomized methods (RBSUMM, PDMM1 and PDMM2) choose the blocks cyclically instead of randomly. Table 5.2 compares the 'best' results of PDMM with other ADMM methods. In PDMM, $(\tau_i, \nu_i) = \{(\frac{1}{2}, 0), (\frac{1}{3}, \frac{1}{2}), (\frac{1}{2}, \frac{1}{2})\}$. GSADMM converges with the smallest number of iterations, but PDMMs can converge faster than GSADMM in terms of runtime. Since GSADMM

[2] http://www.stanford.edu/ boyd/papers/prox_algs/matrix_decomp.html

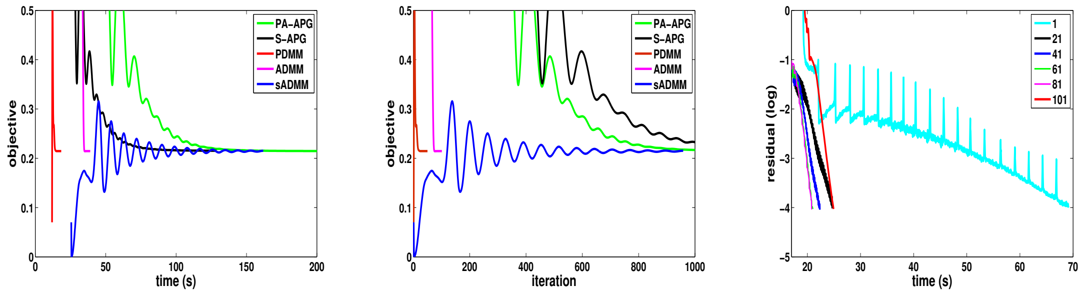


Figure 5.2: Comparison of convergence of PDMM and other methods in overlapping group Lasso.

uses new iterates which increases computation compared to PDMM3, PDMM3 can be faster than GSADMM if the numbers of iterations are close. PDMM1 and PDMM2 can be faster than PDMM3. By simply updating one block, PDMM1 is the fastest algorithm and achieves the lowest residual.

Overlapping Group Lasso: We consider solving the overlapping group lasso problem [222]:

$$\min_{\mathbf{w}} \ \frac{1}{2L\lambda}\|\mathbf{A}\mathbf{w}-\mathbf{b}\|_2^2 + \sum_{g\in\mathcal{G}} d_g\|\mathbf{w}_g\|_2 \ . \tag{5.31}$$

where $\mathbf{A}\in\mathbb{R}^{m\times n}$, $\mathbf{w}\in\mathbb{R}^{n\times 1}$ and $\mathbf{w}_g\in\mathbb{R}^{b\times 1}$ is the vector of overlapping group indexed by g. d_g is some positive weight of group $g\in\mathcal{G}$. As shown in Section 5.2.3, (5.31) can be rewritten as the form (5.15). The data is generated in a same way as [216, 34]: the elements of $\mathbf{A}$ are sampled from normal distribution, $b = \mathbf{A}\mathbf{x}+\epsilon$ with noise ϵ sampled from normal distribution, and $\mathbf{x}_j = (-1)^j\exp(-(j-1)/100)$. In this experiment, $m = 5000$, the number of groups is $L = 100$, and $d_g = \frac{1}{L}, \lambda = \frac{L}{5}$ in (5.31). The size of each group is 100 and the overlap is 10. The total number of blocks in PDMM and sADMM is $J = 101$. τ_i, ν_i in PDMM are computed according to Table (5.1).

In Figure 5.2, the first two figures plot the convergence of objective in terms of the number of iterations and time. PDMM uses all 101 blocks and is the fastest algorithm. ADMM is the same as GSADMM in this problem, but is slower than PDMM. Since sADMM does not consider the sparsity, it uses $\tau_i = \frac{1}{J+1}, \nu_i = 1-\frac{1}{J+1}$, leading to slow convergence. The two accelerated methods, PA-APG [216] and S-APG [34], are slower than PDMM and ADMM.

The effect of K: The third figure shows PDMM with different number of blocks K. Although the complexity of each iteration is the lowest when $K = 1$, PDMM takes much more iterations than other cases and thus takes the longest time. As K increases, PDMM converges faster and faster. When $K = 20$, the runtime is already same as using all blocks. When $K > 21$, PDMM takes less time to converge than using all blocks. The runtime of PDMM decreases as K increases from 21 to 61. However, the speedup from 61 to 81 is negligable. We tried different set of parameters for RBSUMM $\rho\frac{i^2+1}{i+t}(0 \leq i \leq 5, \rho = 0.01, 0.1, 1)$ or sufficiently small step size, but did not see the convergence of the objective within 5000 iterations. Therefore, the results are not included here.

Appexdix

5.A Convergence of PDMM

5.A.1 Technical Preliminaries

We first define some notations will be used specifically in this section. Let $\mathbf{z}_{ij} = \mathbf{A}_{ij}\mathbf{x}_j \in \mathbb{R}^{m_i \times 1}$, $\mathbf{z}_i^r = [\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{iJ}^T]^T \in \mathbb{R}^{m_i J \times 1}$ and $\mathbf{z} = [(\mathbf{z}_1^r)^T, \cdots, (\mathbf{z}_I^r)^T]^T \in \mathbb{R}^{Jm \times 1}$. Let $\mathbf{W}_i \in \mathbb{R}^{Jm_i \times m_i}$ be a column vector of $\mathbf{W}_{ij} \in \mathbb{R}^{m_i \times m_i}$ where

$$\mathbf{W}_{ij} = \begin{cases} \mathbf{I}_{m_i}\ , & \text{if } \mathbf{A}_{ij} \neq \mathbf{0}\ , \\ \mathbf{0} & \text{otherwise}\ . \end{cases} \tag{5.32}$$

Define $\mathbf{Q} \in \mathbb{R}^{Jm \times Jm}$ as a diagonal matrix of $\mathbf{Q}_i \in \mathbb{R}^{Jm_i \times Jm_i}$ and

$$\mathbf{Q} = \text{diag}([\mathbf{Q}_1, \cdots, \mathbf{Q}_I])\ , \mathbf{Q}_i = \text{diag}(\mathbf{W}_i) - \frac{1}{d_i}\mathbf{W}_i\mathbf{W}_i^T\ . \tag{5.33}$$

Therefore, for an optimal solution $\mathbf{x}^*$ satisfying $\mathbf{A}\mathbf{x}^* = \mathbf{a}$, we have

$$\begin{aligned}\|\mathbf{z}^t - \mathbf{z}^*\|_{\mathbf{Q}}^2 &= \sum_{i=1}^I \|\mathbf{z}_i^t - \mathbf{z}_i^*\|_{\mathbf{Q}_i}^2 = \sum_{i=1}^I \|\mathbf{z}_i^t - \mathbf{z}_i^*\|^2_{\text{diag}(\mathbf{w}_i) - \frac{1}{d_i}\mathbf{w}_i\mathbf{w}_i^T} \\ &= \sum_{i=1}^I \left[\sum_{j \in \mathcal{N}(i)} \|\mathbf{z}_{ij}^t - \mathbf{z}_{ij}^*\|_2^2 - \frac{1}{d_i}\|\mathbf{w}_i^T(\mathbf{z}_i^t - \mathbf{z}_i^*)\|_2^2 \right] \\ &= \sum_{i=1}^I \left[\|\mathbf{z}_i^t - \mathbf{z}_i^*\|_2^2 - \frac{1}{d_i}\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2 \right]\ ,\end{aligned} \tag{5.34}$$

where the last equality uses $\mathbf{w}_i^T \mathbf{z}_i^* = \mathbf{A}_i^r \mathbf{x}^* = \mathbf{a}_i$.

In the following lemma, we prove that $\mathbf{Q}_i$ is a positive semi-definite matrix. Thus, $\mathbf{Q}$ is also positive semi-definite.

Lemma 14 $\mathbf{Q}_i$ *is positive semi-definite.*

Proof: As $\mathbf{W}_{ij}$ is either an identity matrix or a zero matrix, $\mathbf{W}_i$ has d_i nonzero entries. Removing the zero entries from $\mathbf{W}_i$, we have $\tilde{\mathbf{W}}_i$ which only has d_i nonzero entries. Then,

$$\tilde{\mathbf{W}}_i = \begin{bmatrix} \mathbf{I}_{m_i} \\ \vdots \\ \mathbf{I}_{m_i} \end{bmatrix}, \text{diag}(\tilde{\mathbf{W}}_i) = \begin{bmatrix} \mathbf{I}_{m_i} & & \\ & \ddots & \\ & & \mathbf{I}_{m_i} \end{bmatrix}, \tag{5.35}$$

diag$(\mathbf{W}_i)$ is an identity matrix. Define $\tilde{\mathbf{Q}}_i = \text{diag}(\tilde{\mathbf{W}}_i) - \frac{1}{d_i}\tilde{\mathbf{W}}_i\tilde{\mathbf{W}}_i^T$. If $\tilde{\mathbf{Q}}_i$ is positive semi-definite, $\mathbf{Q}_i$ is positive semi-definite.

Denote $\lambda_{\tilde{\mathbf{W}}_i}^{\max}$ as the largest eigenvalue of $\tilde{\mathbf{W}}_i\tilde{\mathbf{W}}_i^T$, which is equivalent to the largest eigenvalue of $\tilde{\mathbf{W}}_i^T\tilde{\mathbf{W}}_i$. Since $\tilde{\mathbf{W}}_i^T\tilde{\mathbf{W}}_i = d_i\mathbf{I}_{m_i}$, then $\lambda_{\tilde{\mathbf{W}}_i}^{\max} = d_i$. Then, for any $\mathbf{v}$,

$$\|\mathbf{v}\|^2_{\tilde{\mathbf{W}}_i\tilde{\mathbf{W}}_i^T} \le \lambda_{\tilde{\mathbf{W}}_i}^{\max}\|\mathbf{v}\|_2^2 = d_i\|\mathbf{v}\|_2^2 \,. \tag{5.36}$$

Thus,

$$\|\mathbf{v}\|^2_{\mathbf{Q}_i} = \|\mathbf{v}\|^2_{\text{diag}(\tilde{\mathbf{W}}_i) - \frac{1}{d_i}\tilde{\mathbf{W}}_i\tilde{\mathbf{W}}_i^T} = \|\mathbf{v}\|_2^2 - \frac{1}{d_i}\|\mathbf{v}\|^2_{\tilde{\mathbf{W}}_i\tilde{\mathbf{W}}_i^T} \ge 0\,, \tag{5.37}$$

which completes the proof. ■

Let $\mathbf{W}_i^t \in \mathbb{R}^{Jm_i \times m_i}$ be a column vector of $\mathbf{W}_{ij_t} \in \mathbb{R}^{m_i \times m_i}$ where

$$\mathbf{W}_{ij_t} = \begin{cases} \mathbf{I}_{m_i}\,, & \text{if } \mathbf{A}_{ij_t} \neq \mathbf{0} \text{ and } j_t \in \mathbb{J}_t\,, \\ \mathbf{0} & \text{otherwise}\,. \end{cases} \tag{5.38}$$

Define $\mathbf{P}_t \in \mathbb{R}^{Jm \times Jm}$ as a diagonal matrix of $\mathbf{P}_i^t \in \mathbb{R}^{Jm_i \times Jm_i}$ and

$$\mathbf{P}_t = \text{diag}[\mathbf{P}_1^t, \cdots, \mathbf{P}_I^t]\,, \mathbf{P}_i^t = \text{diag}(\mathbf{W}_i^t) - \frac{1}{\tilde{K}_i}\mathbf{W}_i^t(\mathbf{W}_i^t)^T\,. \tag{5.39}$$

where $\tilde{K}_i = \min\{K, d_i\} \ge \min\{|\mathbb{J}_t \cap \mathcal{N}_i|, d_i\}$. Using similar arguments in Lemma 14, we can show $\mathbf{P}_t$ is positive semi-definite. Therefore,

$$\|\mathbf{z}^{t+1} - \mathbf{z}^t\|^2_{\mathbf{P}_t} = \sum_{i=1}^I \|\mathbf{z}_i^{t+1} - \mathbf{z}_i^t\|^2_{\mathbf{P}_i^t} = \sum_{i=1}^I \|\mathbf{z}_i^{t+1} - \mathbf{z}_i^t\|^2_{\text{diag}(\mathbf{w}_i^t) - \frac{1}{\tilde{K}_i}\mathbf{w}_i^t(\mathbf{w}_i^t)^T}$$

$$= \sum_{i=1}^{I} \left[\sum_{j_t \in \mathbb{J}_t} \|\mathbf{z}_{ij_t}^{t+1} - \mathbf{z}_{ij_t}^t\|_2^2 - \frac{1}{\tilde{K}_i} \|(\mathbf{w}_i^t)^T (\mathbf{z}_i^{t+1} - \mathbf{z}_i^t)\|_2^2 \right]$$

$$= \sum_{i=1}^{I} \left[\|\mathbf{z}_i^{t+1} - \mathbf{z}_i^t\|_2^2 - \frac{1}{\tilde{K}_i} \|\mathbf{A}_i^r (\mathbf{x}^{t+1} - \mathbf{x}^t)\|_2^2 \right] . \tag{5.40}$$

In PDMM, two index set $\mathbb{J}_t$ and $\mathbb{I}_t$ are randomly chosen. Conditioned on $\mathbf{x}^t$, $\mathbf{x}^{t+1}$ and $\mathbf{P}_t$ depend on $\mathbb{J}_t$ but are independent of $\mathbb{I}_t$. Conditioned on $\mathbb{J}_t$, $\mathbf{y}^{t+1}$ depends on $\mathbb{I}_t$. $\mathbf{x}^t, \mathbf{y}^t$ are independent of $\mathbb{I}_t, \mathbb{J}_t$. $\mathbf{x}^t, \mathbf{y}^t$ depend on a sequence of observed realization of random variable

$$\xi_{t-1} = \{(\mathbb{I}_1, \mathbb{J}_1), (\mathbb{I}_2, \mathbb{J}_2), \cdots, (\mathbb{I}_{t-1}, \mathbb{J}_{t-1})\} . \tag{5.41}$$

As we do not assume that f_{j_t} is differentiable, we use the subgradient of f_{j_t}. In particular, if f_{j_t} is differentiable, the subgradient of f_{j_t} becomes the gradient, i.e., $\nabla f_{j_t}(\mathbf{x}_{j_t})$. PDMM (5.5)-(5.7) has the following lemma.

Lemma 15 *Let $\{\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t\}$ be generated by PDMM (5.5)-(5.7). Assume $\tau_i > 0$ and $\nu_i \geq 0$. We have*

$$\begin{aligned}
\sum_{j_t \in \mathbb{J}_t} f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^*) \leq & -\frac{K}{J} \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - (\frac{K_I}{I} - \frac{1}{2}) \tau_i \rho \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right\} \\
& - \sum_{j_t \in \mathbb{I}_t} \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c (\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*) \rangle + \frac{K}{J} \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a} \rangle \\
& + \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - \frac{\tau_i K_I \rho}{2I} \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right\} \\
& - \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^{t+1}, \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \rangle - \frac{\tau_i K_I \rho}{2I} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \right\} \\
& + \frac{\rho}{2} (\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2) \\
& + \sum_{j_t \in \mathbb{J}_t} \eta_{j_t} (B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^t) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^{t+1}) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{j_t}^t)) \\
& + \frac{\rho}{2} \sum_{i=1}^{I} \left\{ [(1 - \frac{2K}{J})(1 - \nu_i) + [(1 - \frac{K}{J}) \frac{K_I}{I} + (1 - \frac{K_I}{I}) \frac{K}{J}] \tau_i + \frac{1}{d_i}] \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right. \\
& \left. - [1 - \nu_i - \frac{K_I}{I} \tau_i + \frac{1}{d_i}] \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 + (1 - \nu_i - \frac{1}{\tilde{K}_i}) \|\mathbf{A}_i^r (\mathbf{x}^{t+1} - \mathbf{x}^t)\|_2^2 \right\} .
\end{aligned} \tag{5.42}$$

Proof: Let $\partial f_{j_t}(\mathbf{x}_{j_t}^{t+1})$ be the subdifferential of f_{j_t} at $\mathbf{x}_{j_t}^{t+1}$. The optimality of the $\mathbf{x}_{j_t}$ update (5.6) is

$$0 \in \partial f_{j_t}(\mathbf{x}_{j_t}^{t+1}) + (\mathbf{A}_{j_t}^c)^T[\hat{\mathbf{y}}^t + \rho(\mathbf{A}_{j_t}^c\mathbf{x}_{j_t}^{t+1} + \sum_{k\neq j_t}\mathbf{A}_k^c\mathbf{x}_k^t - \mathbf{a})] + \eta_{j_t}(\nabla\phi_{j_t}(\mathbf{x}_{j_t}^{t+1}) - \nabla\phi_{j_t}(\mathbf{x}_{j_t}^t)) \ , \tag{5.43}$$

Using (5.5) and rearranging the terms yield

$$-(\mathbf{A}_{j_t}^c)^T[\hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}) + \rho\mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)] + \eta_{j_t}(\nabla\phi_{j_t}(\mathbf{x}_{j_t}^{t+1}) - \nabla\phi_{j_t}(\mathbf{x}_{j_t}^t)) \in \partial f_{j_t}(\mathbf{x}_{j_t}^{t+1}) \ . \tag{5.44}$$

Using the convexity of f_{j_t}, we have

$$\begin{aligned}
& f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^*) \leq -\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^*)\rangle \\
& - \rho\langle \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^*)\rangle - \eta_{j_t}\langle \nabla\phi_{j_t}(\mathbf{x}_{j_t}^{t+1}) - \nabla\phi_{j_t}(\mathbf{x}_{j_t}^t), \mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^*\rangle \\
& = -\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*)\rangle - \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)\rangle \\
& - \rho\sum_{i=1}^I \langle \mathbf{A}_{ij_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t), \mathbf{A}_{ij_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^*)\rangle \\
& + \eta_{j_t}\left(B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^t) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^{t+1}) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{j_t}^t)\right) \ .
\end{aligned} \tag{5.45}$$

Summing over $j_t \in \mathbb{I}_t$, we have

$$\begin{aligned}
& \sum_{j_t\in\mathbb{I}_t} f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^*) \\
& \leq -\sum_{j_t\in\mathbb{I}_t}\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*)\rangle - \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \sum_{j_t\in\mathbb{I}_t}\mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t)\rangle \\
& - \rho\sum_{i=1}^I\sum_{j_t\in\mathbb{I}_t}\langle \mathbf{A}_{ij_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^t), \mathbf{A}_{ij_t}(\mathbf{x}_{j_t}^{t+1} - \mathbf{x}_{j_t}^*)\rangle \\
& + \sum_{j_t\in\mathbb{I}_t}\eta_{j_t}\left(B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^t) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^{t+1}) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{j_t}^t)\right) \\
& = -\sum_{j_t\in\mathbb{I}_t}\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*)\rangle + \frac{K}{J}\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a}\rangle \\
& \underbrace{-\frac{K}{J}\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a}\rangle - \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}(\mathbf{x}^{t+1} - \mathbf{x}^t)\rangle}_{H_1}
\end{aligned}$$

$$+ \underbrace{\frac{\rho}{2}\sum_{i=1}^{I}\sum_{j_t\in\mathbb{I}_t}(\|\mathbf{A}_{ij_t}(\mathbf{x}^*_{j_t}-\mathbf{x}^t_{j_t})\|_2^2 - \|\mathbf{A}_{ij_t}(\mathbf{x}^*_{j_t}-\mathbf{x}^{t+1}_{j_t})\|_2^2 - \|\mathbf{A}_{ij_t}(\mathbf{x}^{t+1}_{j_t}-\mathbf{x}^t_{j_t})\|_2^2)}_{H_2}$$

$$+ \sum_{j_t\in\mathbb{I}_t}\eta_{j_t}\left(B_{\phi_{j_t}}(\mathbf{x}^*_{j_t},\mathbf{x}^t_{j_t}) - B_{\phi_{j_t}}(\mathbf{x}^*_{j_t},\mathbf{x}^{t+1}_{j_t}) - B_{\phi_{j_t}}(\mathbf{x}^{t+1}_{j_t},\mathbf{x}^t_{j_t})\right)\ . \tag{5.46}$$

H_1 in (5.46) can be rewritten as

$$H_1 = -\langle\hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + (1-\frac{K}{J})\langle\hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a}\rangle\ . \tag{5.47}$$

According to (5.7), we have

$$\begin{aligned} &-\langle\mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = -\langle\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \langle\mathbf{y}^{t+1} - \mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle \\ &= -\langle\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \sum_{i_t\in\mathbb{I}_t}\tau_{i_t}\rho\|\mathbf{A}^r_{i_t}\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\|_2^2\ . \end{aligned} \tag{5.48}$$

Taking expectation over $\mathbb{I}_t$, we have

$$-\langle\mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = -\mathbb{E}_{\mathbb{I}_t}\langle\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \frac{K_I}{I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\ . \tag{5.49}$$

The first term of (5.47) is equivalent to

$$\begin{aligned} &-\langle\hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle \\ &= -\sum_{i=1}^{I}\langle\hat{\mathbf{y}}^t_i + \rho(\mathbf{A}^r_i\mathbf{x}^t - \mathbf{a}_i), \mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\rangle \\ &= -\sum_{i=1}^{I}\langle\mathbf{y}^t_i + (1-\nu_i)\rho(\mathbf{A}^r_i\mathbf{x}^t - \mathbf{a}_i), \mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\rangle \\ &= -\langle\mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle - \sum_{i=1}^{I}(1-\nu_i)\rho\langle\mathbf{A}^r_i\mathbf{x}^t - \mathbf{a}_i, \mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\rangle \\ &= -\mathbb{E}_{\mathbb{I}_t}\langle\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \frac{K_I}{I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \\ &+ \sum_{i=1}^{I}\frac{(1-\nu_i)\rho}{2}(\|\mathbf{A}^r_i(\mathbf{x}^{t+1} - \mathbf{x}^t)\|_2^2 - \|\mathbf{A}^r_i\mathbf{x}^t - \mathbf{a}_i\|_2^2 - \|\mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2) \\ &= -\mathbb{E}_{\mathbb{I}_t}\langle\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \frac{K_I}{2I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}^r_i\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \end{aligned}$$

$$+\sum_{i=1}^{I}\left\{\frac{(1-\nu_i)\rho}{2}(\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2)-\frac{(1-\nu_i-\frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\right\}. \tag{5.50}$$

The second term of (5.47) is equivalent to

$$\begin{aligned}&(1-\frac{K}{J})\langle\hat{\mathbf{y}}^t+\rho(\mathbf{A}\mathbf{x}^t-\mathbf{a}),\mathbf{A}\mathbf{x}^t-\mathbf{a}\rangle\\
&=(1-\frac{K}{J})\sum_{i=1}^{I}\langle\hat{\mathbf{y}}_i^t+\rho(\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i),\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle\\
&=(1-\frac{K}{J})\sum_{i=1}^{I}\langle\mathbf{y}_i^t+(1-\nu_i)\rho(\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i),\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle\\
&=(1-\frac{K}{J})\sum_{i=1}^{I}\left\{\langle\mathbf{y}_i^t,\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle-\frac{K_I\tau_i\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&+(1-\frac{K}{J})\sum_{i=1}^{I}(1-\nu_i+\frac{K_I\tau_i}{2I})\rho\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2.\end{aligned} \tag{5.51}$$

H_2 in (5.46) is equavilant to

$$\begin{aligned}H_2&=\frac{\rho}{2}\sum_{i=1}^{I}\sum_{j_t\in\mathbb{I}_t}(\|\mathbf{z}_{ij_t}^*-\mathbf{z}_{ij_t}^t\|_2^2-\|\mathbf{z}_{ij_t}^*-\mathbf{z}_{ij_t}^{t+1}\|_2^2-\|\mathbf{z}_{ij_t}^{t+1}-\mathbf{z}_{ij_t}^t\|_2^2)\\
&=\frac{\rho}{2}\sum_{i=1}^{I}(\|\mathbf{z}_i^*-\mathbf{z}_i^t\|_2^2-\|\mathbf{z}_i^*-\mathbf{z}_i^{t+1}\|_2^2-\|\mathbf{z}_i^{t+1}-\mathbf{z}_i^t\|_2^2)\\
&=\frac{\rho}{2}(\|\mathbf{z}^*-\mathbf{z}^t\|_{\mathbf{Q}}^2-\|\mathbf{z}^*-\mathbf{z}^{t+1}\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{P}_t}^2)\\
&+\frac{\rho}{2}\sum_{i=1}^{I}\frac{1}{d_i}(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)-\frac{1}{\tilde{K}_i}\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2.\end{aligned} \tag{5.52}$$

where the last equality uses the definition of $\mathbf{Q}$ in (5.33) and $\mathbf{P}_t$ (5.39), and $\tilde{K}_i=\min\{K,d_i\}$. Combining the results of (5.47)-(5.52) gives

$$\begin{aligned}&H_1+H_2=-\mathbb{E}_{\mathbb{I}_t}\langle\mathbf{y}^{t+1},\mathbf{A}\mathbf{x}^{t+1}-\mathbf{a}\rangle+\frac{K_I}{2I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\\
&+\sum_{i=1}^{I}\left\{\frac{(1-\nu_i)\rho}{2}(\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2)-\frac{(1-\nu_i-\frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\right\}\end{aligned}$$

$$
\begin{aligned}
&+(1-\frac{K}{J})\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle - \frac{K_I\tau_i\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&+(1-\frac{K}{J})\sum_{i=1}^{I}(1-\nu_i+\frac{K_I\tau_i}{2I})\rho\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\\
&+\frac{\rho}{2}(\|\mathbf{z}^*-\mathbf{z}^t\|_{\mathbf{Q}}^2-\|\mathbf{z}^*-\mathbf{z}^{t+1}\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{P}_t}^2)\\
&+\frac{\rho}{2}\sum_{i=1}^{I}\frac{1}{d_i}(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)-\frac{1}{\tilde{K}_i}\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2)\\
&=-\frac{K}{J}\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle-(\frac{K_I}{I}-\frac{1}{2})\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&+\left\{\langle \mathbf{y}^t, \mathbf{A}\mathbf{x}^t-\mathbf{a}\rangle-\sum_{i=1}^{I}\frac{K_I\tau_i\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&-\left\{\mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1}-\mathbf{a}\rangle-\sum_{i=1}^{I}\frac{K_I\tau_i\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\right\}\\
&+\frac{\rho}{2}(\|\mathbf{z}^*-\mathbf{z}^t\|_{\mathbf{Q}}^2-\|\mathbf{z}^*-\mathbf{z}^{t+1}\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{P}_t}^2)\\
&+\frac{\rho}{2}\sum_{i=1}^{I}\left\{[(1-\frac{2K}{J})(1-\nu_i)+[(1-\frac{K}{J})\frac{K_I}{I}+(1-\frac{K_I}{I})\frac{K}{J}]\tau_i+\frac{1}{d_i}]\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right.\\
&\left.-(1-\nu_i-\frac{K_I}{I}\tau_i+\frac{1}{d_i})\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2+(1-\nu_i-\frac{1}{\tilde{K}_i})\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2\right\}. \qquad (5.53)
\end{aligned}
$$

Plugging back into (5.46) completes the proof. ■

Lemma 16 *Let $\{\mathbf{x}_{j_t}^t, \mathbf{y}_i^t\}$ be generated by PDMM (5.5)-(5.7). Assume $\tau_i > 0$ and $\nu_i \geq 0$. We have*

$$
\begin{aligned}
&\sum_{j_t\in\mathbb{J}_t} f_{j_t}(\mathbf{x}_{j_t}^{t+1})-f_{j_t}(\mathbf{x}_{j_t}^*)\leq -\frac{K}{J}\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle-(\frac{K_I}{I}-\frac{1}{2})\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&-\sum_{j_t\in\mathbb{I}_t}\langle \hat{\mathbf{y}}^t+\rho(\mathbf{A}\mathbf{x}^t-\mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^t-\mathbf{x}_{j_t}^*)\rangle+\frac{K}{J}\langle \hat{\mathbf{y}}^t+\rho(\mathbf{A}\mathbf{x}^t-\mathbf{a}), \mathbf{A}\mathbf{x}^t-\mathbf{a}\rangle\\
&+\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\rangle-\frac{\tau_iK_I\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2\right\}\\
&-\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^{t+1}, \mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\rangle-\frac{\tau_iK_I\rho}{2I}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\right\}
\end{aligned}
$$

$$+\frac{\rho}{2}(\|\mathbf{z}^*-\mathbf{z}^t\|_{\mathbf{Q}}^2-\|\mathbf{z}^*-\mathbf{z}^{t+1}\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{P}_t}^2)$$
$$+\boldsymbol{\eta}^T(B_{\boldsymbol{\phi}}(\mathbf{x}^*,\mathbf{x}^t)-B_{\phi}(\mathbf{x}^*,\mathbf{x}^{t+1})-B_{\phi}(\mathbf{x}^{t+1},\mathbf{x}^t))$$
$$+\frac{\rho}{2}\sum_{i=1}^{I}\left[\gamma_i(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)-\beta_i\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\right]\ . \tag{5.54}$$

where $\boldsymbol{\eta}^T=[\eta_1,\cdots,\eta_J]$. $\tau_i>0,\nu_i\geq 0,\gamma_i\geq 0$ *and* $\beta_i\geq 0$ *satisfy the following conditions:*

$$\nu_i\in(\max\{0,1-\frac{2J}{\tilde{K}_i(2J-K)}\},1-\frac{1}{\tilde{K}_i}]\ , \tag{5.55}$$

$$\tau_i\leq\frac{J}{(2J-K)\frac{K_I}{I}+K(1-\frac{K_I}{I})}[\frac{4}{\tilde{K}_i}-(4-\frac{2K}{J})(1-\nu_i)]\leq\frac{2K}{\tilde{K}_i[(2J-K)\frac{K_I}{I}+K(1-\frac{K_I}{I})]}\ , \tag{5.56}$$

$$\gamma_i=(3-\frac{2K}{J})(1-\nu_i)+\left[(1-\frac{K}{J})\frac{K_I}{I}+\frac{K}{J}(1-\frac{K_I}{I})\right]\tau_i+\frac{1}{d_i}-\frac{2}{\tilde{K}_i}\ , \tag{5.57}$$

$$\beta_i=\frac{4}{\tilde{K}_i}-2(2-\frac{K}{J})(1-\nu_i)-\left[(2-\frac{K}{J})\frac{K_I}{I}+\frac{K}{J}(1-\frac{K_I}{I})\right]\tau_i\ . \tag{5.58}$$

Proof: In (5.42), denote

$$H_3=[(1-\frac{2K}{J})(1-\nu_i)+[(1-\frac{K}{J})\frac{K_I}{I}+(1-\frac{K_I}{I})\frac{K}{J}]\tau_i+\frac{1}{d_i}]\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2$$
$$-(1-\nu_i-\frac{K_I}{I}\tau_i+\frac{1}{d_i})\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\ , \tag{5.59}$$

$$H_4=(1-\nu_i-\frac{1}{\tilde{K}_i})\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2\ . \tag{5.60}$$

Our goal is to eliminate H_4 so that

$$H_3+H_4=\gamma_i(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)-\beta_i\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\ , \tag{5.61}$$

where $\gamma_i\geq 0$ and $\beta_i\geq 0$.

We want to choose a large τ_i and a small ν_i. Assume $1-\nu_i-\frac{1}{\tilde{K}_i}\geq 0$, i.e., $\nu_i\leq 1-\frac{1}{\tilde{K}_i}$, we have

$$H_4=(1-\nu_i-\frac{1}{\tilde{K}_i})\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2\leq 2(1-\nu_i-\frac{1}{\tilde{K}_i})(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2+\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)\ . \tag{5.62}$$

Therefore, we have

$$H_3+H_4\leq[(3-\frac{2K}{J})(1-\nu_i)+[(1-\frac{K}{J})\frac{K_I}{I}+(1-\frac{K_I}{I})\frac{K}{J}]\tau_i+\frac{1}{d_i}-\frac{2}{\tilde{K}_i}]\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2$$

$$+ (1 - \nu_i + \frac{K_I}{I}\tau_i - \frac{1}{d_i} - \frac{2}{\tilde{K}_i})\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2$$
$$= \gamma_i(\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2 - \|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2) - \beta_i\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,. \tag{5.63}$$

where

$$\begin{aligned}\gamma_i &= (3 - \frac{2K}{J})(1 - \nu_i) + \left[(1 - \frac{K}{J})\frac{K_I}{I} + \frac{K}{J}(1 - \frac{K_I}{I})\right]\tau_i + \frac{1}{d_i} - \frac{2}{\tilde{K}_i} \\ &\geq (3 - \frac{2K}{J})\frac{1}{\tilde{K}_i} + \left[(1 - \frac{K}{J})\frac{K_I}{I} + \frac{K}{J}(1 - \frac{K_I}{I})\right]\tau_i + \frac{1}{d_i} - \frac{2}{\tilde{K}_i} \\ &= (1 - \frac{K}{J})\frac{1}{\tilde{K}_i} - \frac{K}{J\tilde{K}_i} + \frac{1}{d_i} + \left[(1 - \frac{K}{J})\frac{K_I}{I} + \frac{K}{J}(1 - \frac{K_I}{I})\right]\tau_i \geq 0 \,.\end{aligned} \tag{5.64}$$

and

$$\begin{aligned}\beta_i &= -(1 - \nu_i + \frac{K_I}{I}\tau_i + \frac{1}{d_i} - \frac{2}{\tilde{K}_i} + \gamma_i) \\ &= \frac{4}{\tilde{K}_i} - 2(2 - \frac{K}{J})(1 - \nu_i) - \left[(2 - \frac{K}{J})\frac{K_I}{I} + \frac{K}{J}(1 - \frac{K_I}{I})\right]\tau_i \,.\end{aligned} \tag{5.65}$$

We also want $\beta_i \geq 0$, which can be reduced to

$$\begin{aligned}\tau_i &\leq \frac{J}{(2J - K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})}[\frac{4}{\tilde{K}_i} - (4 - \frac{2K}{J})(1 - \nu_i)] \\ &\leq \frac{J}{(2J - K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})}[\frac{4}{\tilde{K}_i} - (4 - \frac{2K}{J})\frac{1}{\tilde{K}_i}] \\ &= \frac{2K}{\tilde{K}_i[(2J - K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})]} \,.\end{aligned} \tag{5.66}$$

It also requires the RHS of (5.66) to be positive, leading to $\nu_i > \max\{0, 1 - \frac{2J}{\tilde{K}_i(2J-K)}\}$. Therefore, $\nu_i \in (\max\{0, 1 - \frac{2J}{\tilde{K}_i(2J-K)}\}, 1 - \frac{1}{\tilde{K}_i}]$.

Denote $B_{\boldsymbol{\phi}} = [B_{\phi_1}, \cdots, B_{\boldsymbol{\phi}_J}]^T$ as a column vector of the Bregman divergence on block coordinates of $\mathbf{x}$. Using $\mathbf{x}^{t+1} = [\mathbf{x}_{j_t\in\mathbb{I}_t}^{t+1}, \mathbf{x}_{j_t\notin\mathbb{I}_t}^t]^T$, we have $B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^t) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^{t+1}) = B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - B_{\phi}(\mathbf{x}^*, \mathbf{x}^{t+1}), B_{\phi_{j_t}}(\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{j_t}^t) = B_{\phi}(\mathbf{x}^{t+1}, \mathbf{x}^t)$. Thus,

$$\begin{aligned}&\sum_{j_t\in\mathbb{I}_t}\eta_{j_t}\left(B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^t) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^*, \mathbf{x}_{j_t}^{t+1}) - B_{\phi_{j_t}}(\mathbf{x}_{j_t}^{t+1}, \mathbf{x}_{j_t}^t)\right) \\ &= \boldsymbol{\eta}^T(B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \,.\end{aligned} \tag{5.67}$$

where $\boldsymbol{\eta}^T = [\eta_1, \cdots, \eta_J]$. ■

Lemma 17 *Let $\{\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t\}$ be generated by PDMM (5.5)-(5.7). Assume $\tau_i > 0$ and $\nu_i \geq 0$ satisfy the conditions in Lemma 16. We have*

$$\begin{aligned} f(\mathbf{x}^t) - f(\mathbf{x}^*) \leq & \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - (\frac{K_I}{I} - \frac{1}{2}) \tau_i \rho \| \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \|_2^2 \right\} \\ & + \frac{J}{K} \Bigg\{ \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\mathbb{I}_t} \tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2} \sum_{i=1}^{I} \beta_i \mathbb{E}_{\mathbb{I}_t} \| \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \|_2^2 \\ & + \frac{\rho}{2} (\| \mathbf{z}^* - \mathbf{z}^t \|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t} \| \mathbf{z}^* - \mathbf{z}^{t+1} \|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t} \| \mathbf{z}^{t+1} - \mathbf{z}^t \|_{\mathbf{P}_t}^2) \\ & + \boldsymbol{\eta}^T (B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \Bigg\} . \end{aligned} \tag{5.68}$$

where $\tilde{\mathcal{L}}_\rho$ is defined as follows:

$$\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) = f(\mathbf{x}^t) - f(\mathbf{x}^*) + \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle + \frac{(\gamma_i - \frac{K_I}{I} \tau_i)\rho}{2} \| \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \|_2^2 \right\} . \tag{5.69}$$

$\tau_i, \nu_i, \gamma_i, \beta_i$ and $\boldsymbol{\eta}$ are defined in Lemma 16.

Proof: Using $\mathbf{x}^{t+1} = [\mathbf{x}_{j_t \in \mathbb{J}_t}^{t+1}, \mathbf{x}_{j_t \notin \mathbb{J}_t}^t]^T$, we have

$$\begin{aligned} f(\mathbf{x}^{t+1}) - f(\mathbf{x}^t) &= \sum_{j_t \in \mathbb{J}_t} f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^t) \\ &= \sum_{j_t \in \mathbb{J}_t} [f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^*)] - \sum_{j_t \in \mathbb{J}_t} [f_{j_t}(\mathbf{x}_{j_t}^t) - f_{j_t}(\mathbf{x}_{j_t}^*)] . \end{aligned} \tag{5.70}$$

Rearranging the terms and using Lemma 16 yield

$$\begin{aligned} & \sum_{j_t \in \mathbb{J}_t} f_{j_t}(\mathbf{x}_{j_t}^t) - f_{j_t}(\mathbf{x}_{j_t}^*) = \sum_{j \in \mathbb{J}_t} [f_{j_t}(\mathbf{x}_{j_t}^{t+1}) - f_{j_t}(\mathbf{x}_{j_t}^*)] + f(\mathbf{x}^t) - f(\mathbf{x}^{t+1}) \\ & \leq -\frac{K}{J} \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - (\frac{K_I}{I} - \frac{1}{2}) \tau_i \rho \| \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \|_2^2 \right\} \\ & - \sum_{j_t \in \mathbb{J}_t} \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c (\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*) \rangle + \frac{K}{J} \langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a} \rangle \\ & + \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2} \sum_{i=1}^{I} \beta_i \| \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \|_2^2 \end{aligned}$$

$$+ \frac{\rho}{2}(\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2)$$
$$+ \boldsymbol{\eta}^T(B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - B_{\phi}(\mathbf{x}^*, \mathbf{x}^{t+1}) - B_{\phi}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \ , \tag{5.71}$$

where $\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t)$ is defined in (5.69). Conditioning on $\mathbf{x}^t$ and taking expectation over $\mathbb{J}_t$, we have

$$\frac{K}{J}[f(\mathbf{x}^t) - f(\mathbf{x}^*)] \leq -\frac{K}{J}\sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\rangle - (\frac{K_I}{I} - \frac{1}{2})\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2\right\}$$
$$+ \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\mathbb{I}_t}\tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2}\sum_{i=1}^{I}\beta_i\mathbb{E}_{\mathbb{I}_t}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2$$
$$+ \frac{\rho}{2}(\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2)$$
$$+ \boldsymbol{\eta}^T(B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\mathbb{I}_t}B_{\phi}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\mathbb{I}_t}B_{\phi}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \ , \tag{5.72}$$

where we use

$$\mathbb{E}_{\mathbb{J}_t}[-\sum_{j_t\in\mathbb{J}_t}\langle \hat{\mathbf{y}^t} + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}_{j_t}^c(\mathbf{x}_{j_t}^t - \mathbf{x}_{j_t}^*)\rangle] = -\frac{K}{J}\langle \hat{\mathbf{y}}^t + \rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}), \mathbf{A}\mathbf{x}^t - \mathbf{a}\rangle \ . \tag{5.73}$$

Dividing both sides by $\frac{K}{J}$ and using the definition (5.69) complete the proof. ■

For the randomized dual block coordinate update, we have the following results.

Lemma 18 *Let $\{\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t\}$ be generated by PDMM (5.5)-(5.7). Assume $\tau_i > 0$ and $\nu_i \geq 0$. We have*

$$\mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^* - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = \frac{I}{K_I}\sum_{i=1}^{I}\frac{1}{2\tau_i\rho}\left[\|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^{t+1}\|_2^2\right]$$
$$+ (\frac{1}{2} - \frac{K_I}{I})\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \ . \tag{5.74}$$

Proof: According to (5.7), we have

$$\mathbf{y}^{t+1} - \mathbf{y}^t = \sum_{i_t\in\mathbb{I}_t}(\mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}) \ . \tag{5.75}$$

Using $\mathbf{y}^{t+1} = [\mathbf{y}_{i_t\in\mathbb{I}_t}^{t+1}, \mathbf{y}_{k\notin\mathbb{I}_t}^t]^T$, we have

$$\sum_{i_t\in\mathbb{I}_t}\langle \mathbf{y}_{i_t}^* - \mathbf{y}_{i_t}^t, \mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\rangle$$

$$
\begin{aligned}
&= \sum_{i_t\in\mathbb{I}_t} \left[\langle \mathbf{y}_{i_t}^* - \mathbf{y}_{i_t}^{t+1}, \mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\rangle + \langle \mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}, \mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\rangle\right] \\
&= \sum_{i_t\in\mathbb{I}_t} \frac{1}{\tau_{i_t}\rho}\left[\langle \mathbf{y}_{i_t}^* - \mathbf{y}_{i_t}^{t+1}, \mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}\rangle + \|\mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}\|_2^2\right] \\
&= \sum_{i=1}^{I} \frac{1}{\tau_{i}\rho}\left[\langle \mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}, \mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t}\rangle + \|\mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t}\|_2^2\right] \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_{i}\rho}\left[\|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t}\|_2^2 - \|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}\|_2^2 + \|\mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t}\|_2^2\right] \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_{i}\rho}\left[\|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t}\|_2^2 - \|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}\|_2^2\right] + \sum_{i_t\in\mathbb{I}_t} \frac{1}{2\tau_{i}\rho}\|\mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}\|_2^2 \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_{i}\rho}\left[\|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t}\|_2^2 - \|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}\|_2^2\right] + \sum_{i_t\in\mathbb{I}_t} \frac{\tau_{i_t}\rho}{2}\|\mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\|_2^2\,. \qquad (5.76)
\end{aligned}
$$

$\mathbf{x}^{t+1}$ is independent of $\mathbb{I}_t$. Taking expectation over $\mathbb{I}_t$, we have

$$
\begin{aligned}
\frac{K_I}{I}\sum_{i=1}^{I}\langle \mathbf{y}_i^* - \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\rangle = \sum_{i=1}^{I} \frac{1}{2\tau_{i}\rho}\left[\|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t}\|_2^2 - \|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}\|_2^2\right] \\
+ \frac{K_I}{I}\sum_{i=1}^{I}\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\,. \qquad (5.77)
\end{aligned}
$$

Dividing both sides by $\frac{K_I}{I}$ yields

$$
\langle \mathbf{y}^* - \mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = \frac{I}{K_I}\sum_{i=1}^{I} \frac{1}{2\tau_{i}\rho}\left[\|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t}\|_2^2 - \|\mathbf{y}_{i}^* - \mathbf{y}_{i}^{t+1}\|_2^2\right] + \sum_{i=1}^{I}\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\,. \qquad (5.78)
$$

Using (5.7) and the fact that $\mathbf{y}^{t+1} = [\mathbf{y}_{i_t\in\mathbb{I}_t}^{t+1}, \mathbf{y}_{k\notin\mathbb{I}_t}^{t}]^T$, we have

$$
\langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = \sum_{i_t\in\mathbb{I}_t}\langle \mathbf{y}_{i_t}^t - \mathbf{y}_{i_t}^{t+1}, \mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\rangle = -\sum_{i_t\in\mathbb{I}_t}\tau_{i_t}\rho\|\mathbf{A}_{i_t}^r\mathbf{x}^{t+1} - \mathbf{a}_{i_t}\|_2^2\,. \qquad (5.79)
$$

Taking expectation over $\mathbb{I}_t$, we have

$$
\mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle = -\frac{K_I}{I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\,. \qquad (5.80)
$$

Adding (5.78) and (5.80), we have

$$
\begin{aligned}
&\mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^* - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle \\
&= \mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^* - \mathbf{y}^{t}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle + \mathbb{E}_{\mathbb{I}_t}\langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}\rangle \\
&= \frac{I}{K_I}\sum_{i=1}^{I}\frac{1}{2\tau_i\rho}\left[\|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^{t+1}\|_2^2\right] \\
&+ \sum_{i=1}^{I}\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 - \frac{K_I}{I}\sum_{i=1}^{I}\tau_i\rho\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,, \qquad (5.81)
\end{aligned}
$$

which completes the proof. ■

5.A.2 Theoretical Results

We establish the convergence results for PDMM under fairly simple assumptions:

Assumption 8

(1) $f_j : \mathbb{R}^{n_j} \to \mathbb{R} \cup \{+\infty\}$ *are closed, proper, and convex.*

(2) A KKT point of the Lagrangian ($\rho = 0$ *in (5.2)) of Problem (5.1) exists.*

Assumption 8 is the same as that required by ADMM [19, 200]. Let ∂f_j be the subdifferential of f_j. Assume that $\{\mathbf{x}_j^*, \mathbf{y}_i^*\}$ satisfies the KKT conditions of the Lagrangian ($\rho = 0$ in (5.2)), i.e.,

$$
-\mathbf{A}_j^T\mathbf{y}^* \in \partial f_j(\mathbf{x}_j^*) \,, \qquad (5.82)
$$

$$
\mathbf{A}\mathbf{x}^* - \mathbf{a} = 0. \qquad (5.83)
$$

During iterations, (5.83) is satisfied if $\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a}$. The optimality conditions for the $\mathbf{x}_j$ update (5.6) is

$$
0 \in \partial f_j(\mathbf{x}_j^{t+1}) + \mathbf{A}_j^c[\hat{\mathbf{y}}^t + \rho(\mathbf{A}_j^c\mathbf{x}_j^{t+1} + \sum_{k\neq j}\mathbf{A}_k^c\mathbf{x}_k^t - \mathbf{a})] + \eta_j(\nabla\phi_j(\mathbf{x}_j^{t+1}) - \nabla\phi_j(\mathbf{x}_j^t)) \,, \qquad (5.84)
$$

which is equivalent to

$$
-\mathbf{A}_j^c[\mathbf{y}^t + (1-\nu)\rho(\mathbf{A}\mathbf{x}^t - \mathbf{a}) + \mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)] - \eta_j(\nabla\phi_j(\mathbf{x}_j^{t+1}) - \nabla\phi_j(\mathbf{x}_j^t)) \in \partial f_j(\mathbf{x}_j^{t+1}) \,. \qquad (5.85)
$$

When $\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a}, \mathbf{y}^{t+1} = \mathbf{y}^t$. If $\mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t) = 0$, then $\mathbf{A}\mathbf{x}^t - \mathbf{a} = 0$. When $\eta_j \geq 0$, further assuming $B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) = 0$, (5.82) will be satisfied. Overall, the KKT conditions (5.82)-(5.83) are satisfied if the following optimality conditions are satisfied by the iterates:

$$\mathbf{A}\mathbf{x}^{t+1} = \mathbf{a}\ , \mathbf{A}_j^c(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t) = 0\ , \tag{5.86}$$

$$B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) = 0\ . \tag{5.87}$$

The above optimality conditions are sufficient for the KKT conditions. (5.86) are the optimality conditions for the exact PDMM. (5.87) is needed only when $\eta_j > 0$.

In Lemma 16, setting the values of $\nu_i, \tau_i, \gamma_i, \beta_i$ as follows:

$$\nu_i = 1 - \frac{1}{\tilde{K}_i}\ , \tau_i = \frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})]}\ , \tag{5.88}$$

$$\gamma_i = \frac{J-K}{J\tilde{K}_i} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i} + \frac{K[(J-K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})]}{J\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})]}\ , \beta_i = \frac{K}{J\tilde{K}_i}\ . \tag{5.89}$$

Define the residual of optimality conditions (5.86)-(5.87) as

$$R(\mathbf{x}^{t+1}) = \frac{\rho}{2}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2 + \frac{\rho}{2}\sum_{i=1}^{I}\beta_i\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 + \boldsymbol{\eta}^T B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t)\ . \tag{5.90}$$

If $R(\mathbf{x}^{t+1}) \to 0$, (5.86)-(5.87) will be satisfied and thus PDMM converges to the KKT point $\{\mathbf{x}^*, \mathbf{y}^*\}$.

Define the current iterate $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ and $h(\mathbf{v}^*, \mathbf{v}^t)$ as a distance from $\mathbf{v}^t$ to a KKT point $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$:

$$h(\mathbf{v}^*, \mathbf{v}^t) = \frac{K}{J}\sum_{i=1}^{I}\frac{I}{2K_I\tau_i\rho}\|\mathbf{y}_i^* - \mathbf{y}_i^{t-1}\|_2^2 + \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 + \boldsymbol{\eta}^T B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t)\ . \tag{5.91}$$

The following Lemma shows that $h(\mathbf{v}^*, \mathbf{v}^t) \geq 0$.

Lemma 19 *Let $h(\mathbf{v}^*, \mathbf{v}^t)$ be defined in (5.91). Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})]}$, we have*

$$h(\mathbf{v}^*, \mathbf{v}^t) \geq \frac{\rho}{2}\sum_{i=1}^{I}\zeta_i\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2 + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 + + \sum_{j=1}^{J}\eta_j B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) \geq 0\ . \tag{5.92}$$

where $\zeta_i = \frac{(J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})]} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i} \geq 0$. Moreover, if $h(\mathbf{v}^*, \mathbf{v}^t) = 0$, then $\mathbf{A}_i^r\mathbf{x}^t = \mathbf{a}_i, \mathbf{z}^t = \mathbf{z}^*$ and $B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) = 0$. Thus, (5.82)-(5.83) are satisfied.

Proof: Using the convexity of f and (5.82), we have

$$f(\mathbf{x}^*) - f(\mathbf{x}^t) \leq -\langle \mathbf{A}^T\mathbf{y}^*, \mathbf{x}^* - \mathbf{x}^t\rangle = \sum_{i=1}^{I}\langle \mathbf{y}_i^*, \mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\rangle \; . \tag{5.93}$$

Thus,

$$\begin{aligned}
&\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) = f(\mathbf{x}^t) - f(\mathbf{x}^*) + \sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t, \mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\rangle + \frac{(\gamma_i - \frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2\right\} \\
&\geq \sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^t - \mathbf{y}_i^*, \mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\rangle + \frac{(\gamma_i - \frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2\right\} \\
&= \sum_{i=1}^{I}\left\{\langle \mathbf{y}_i^{t-1} - \mathbf{y}_i^*, \mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\rangle + \langle \mathbf{y}_i^t - \mathbf{y}_i^{t-1}, \mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\rangle + \frac{(\gamma_i - \frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2\right\} \\
&\geq \sum_{i=1}^{I}\left[-\frac{KI}{2JK_I\tau_i\rho}\|\mathbf{y}_i^{t-1} - \mathbf{y}_i^*\|_2^2 - \frac{JK_I\tau_i\rho}{2KI}\|\mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\|_2^2 + \frac{(\gamma_i + \frac{K_I}{I}\tau_i)\rho}{2}\|\mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\|_2^2\right] \\
&= \sum_{i=1}^{I}\left[-\frac{KI}{2JK_I\tau_i\rho}\|\mathbf{y}_i^{t-1} - \mathbf{y}_i^*\|_2^2 + [\gamma_i + (1 - \frac{J}{K})\frac{K_I}{I}\tau_i]\frac{\rho}{2}\|\mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\|_2^2\right] \; .
\end{aligned} \tag{5.94}$$

$h(\mathbf{v}^*, \mathbf{v}^t)$ is reduced to

$$h(\mathbf{v}^*, \mathbf{v}^t) \geq \frac{\rho}{2}\sum_{i=1}^{I}[\gamma_i + (1 - \frac{J}{K})\frac{K_I}{I}\tau_i]\|\mathbf{A}_i\mathbf{x}^t - \mathbf{a}_i\|_2^2 + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 + \boldsymbol{\eta}^T B_\phi(\mathbf{x}^*, \mathbf{x}^t) \; . \tag{5.95}$$

Setting $1 - \nu_i = \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})]}$, we have

$$\begin{aligned}
&\gamma_i + (1 - \frac{J}{K})\tau_i \\
&= (3 - \frac{2K}{J})(1 - \nu_i) + \left[(1 - \frac{K}{J})\frac{K_I}{I} + \frac{K}{J}(1 - \frac{K_I}{I})\right]\tau_i + \frac{1}{d_i} - \frac{2}{\tilde{K}_i} + (1 - \frac{J}{K})\frac{K_I}{I}\tau_i \\
&= \frac{(J-K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1 - \frac{K_I}{I})]} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i} \geq 0 \; .
\end{aligned} \tag{5.96}$$

Therefore, $h(\mathbf{v}^*, \mathbf{v}^t) \geq 0$. Letting $\zeta_i = \frac{(J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})]} + \frac{1}{d_i} - \frac{K}{J\tilde{K}_i}$ completes the proof. ■

The following theorem shows that $h(\mathbf{v}^*, \mathbf{v}^t)$ decreases monotonically and thus establishes the global convergence of PDMM.

Theorem 15 *(Global Convergence of PDMM) Let $\mathbf{v}^t = (\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t)$ be generated by PDMM (5.5)-(5.7) and $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$ be a KKT point satisfying (5.82)-(5.83). Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I}+K(1-\frac{K_I}{I})]}$, we have*

$$0 \le \mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) \le \mathbb{E}_{\xi_{t-1}} h(\mathbf{v}^*, \mathbf{v}^t) \ , \quad \mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \to 0 \ . \tag{5.97}$$

Proof: Adding (5.94) and (5.68) yields

$$\begin{aligned} 0 \le & \sum_{i=1}^I \left\{ \langle \mathbf{y}_i^* - \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - (\frac{K_I}{I} - \frac{1}{2})\tau_i \rho \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right\} \\ & + \frac{J}{K} \Bigg\{ \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\mathbb{I}_t} \tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2}\sum_{i=1}^I \beta_i \mathbb{E}_{\mathbb{I}_t} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \\ & + \frac{\rho}{2}(\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2) \\ & + \boldsymbol{\eta}^T (B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \Bigg\} \ . \end{aligned} \tag{5.98}$$

According to Lemma 18, we have

$$\begin{aligned} & \sum_{i=1}^I \left\{ \langle \mathbf{y}_i^* - \mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i \rangle - (\frac{K_I}{I} - \frac{1}{2})\tau_i \rho \|\mathbf{A}_i^r \mathbf{x}^t - \mathbf{a}_i\|_2^2 \right\} \\ & = \frac{I}{K_I} \sum_{i=1}^I \frac{1}{2\tau_i \rho} \left[\|\mathbf{y}_i^* - \mathbf{y}_i^{t-1}\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 \right] \ . \end{aligned} \tag{5.99}$$

Plugging back into (5.98) gives

$$\begin{aligned} 0 \le & \frac{I}{K_I} \sum_{i=1}^I \frac{1}{2\tau_i \rho} (\|\mathbf{y}_i^* - \mathbf{y}_i^{t-1}\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2) \\ & + \frac{J}{K} \Bigg\{ \tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\mathbb{I}_t} \tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2}\sum_{i=1}^I \beta_i \mathbb{E}_{\mathbb{I}_t} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \\ & + \frac{\rho}{2}(\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2) \\ & + \boldsymbol{\eta}^T (B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\mathbb{I}_t} B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t)) \Bigg\} \\ & = \frac{J}{K} \left\{ h(\mathbf{v}^*, \mathbf{v}^t) - \mathbb{E}_{\mathbb{I}_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) - \mathbb{E}_{\mathbb{I}_t} R(\mathbf{x}^{t+1}) \right\} \ . \end{aligned} \tag{5.100}$$

Taking expectaion over ξ_{t-1}, we have

$$0 \leq \frac{J}{K} \left\{ \mathbb{E}_{\xi_{t-1}} h(\mathbf{v}^*, \mathbf{v}^t) - \mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) - \mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \right\} . \tag{5.101}$$

Since $\mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \geq 0$, we have

$$\mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) \leq \mathbb{E}_{\xi_{t-1}} h(\mathbf{v}^*, \mathbf{v}^t) . \tag{5.102}$$

Thus, $\mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1})$ converges monotonically.

Rearranging the terms in (5.101) yields

$$\mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \leq \mathbb{E}_{\xi_{t-1}} h(\mathbf{v}^*, \mathbf{v}^t) - \mathbb{E}_{\xi_t} h(\mathbf{v}^*, \mathbf{v}^{t+1}) . \tag{5.103}$$

Summing over t gives

$$\sum_{t=0}^{T-1} \mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \leq h(\mathbf{v}^*, \mathbf{v}^0) - \mathbb{E}_{\xi_{T-1}} h(\mathbf{v}^*, \mathbf{v}^T) \leq h(\mathbf{v}^*, \mathbf{v}^0) . \tag{5.104}$$

where the last inequality uses the Lemma 19. As $T \rightarrow \infty$, $\mathbb{E}_{\xi_t} R(\mathbf{x}^{t+1}) \rightarrow 0$, which completes the proof. ∎

Similar as the Lemma 18, we have the following results.

Lemma 20 *Let $\{\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t\}$ be generated by PDMM (5.5)-(5.7). Assume $\tau_i > 0$ and $\nu_i \geq 0$. We have*

$$\begin{aligned} -\mathbb{E}_{\mathbb{I}_t} \langle \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle = & \frac{I}{K_I} \sum_{i=1}^{I} \frac{1}{2\tau_i \rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] \\ & + (\frac{1}{2} - \frac{K_I}{I}) \sum_{i=1}^{I} \tau_i \rho \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 . \end{aligned} \tag{5.105}$$

Proof: According to (5.7), we have

$$\mathbf{y}^{t+1} - \mathbf{y}^t = \sum_{i_t \in \mathbb{I}_t} (\mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t}) . \tag{5.106}$$

Using $\mathbf{y}^{t+1} = [\mathbf{y}_{i_t \in \mathbb{I}_t}^{t+1}, \mathbf{y}_{k \notin \mathbb{I}_t}^t]^T$, we have

$$\sum_{i_t \in \mathbb{I}_t} \langle -\mathbf{y}_{i_t}^t, \mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t} \rangle = \sum_{i_t \in \mathbb{I}_t} \left[\langle -\mathbf{y}_{i_t}^{t+1}, \mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t} \rangle + \langle \mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^t, \mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t} \rangle \right]$$

$$
\begin{aligned}
&= \sum_{i_t \in \mathbb{I}_t} \frac{1}{\tau_{i_t}\rho} \left[\langle -\mathbf{y}_{i_t}^{t+1}, \mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t} \rangle + \|\mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}\|_2^2 \right] \\
&= \sum_{i=1}^{I} \frac{1}{\tau_i\rho} \left[\langle -\mathbf{y}_{i}^{t+1}, \mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t} \rangle + \|\mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t}\|_2^2 \right] \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 + \|\mathbf{y}_{i}^{t+1} - \mathbf{y}_{i}^{t}\|_2^2 \right] \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] + \sum_{i_t \in \mathbb{I}_t} \frac{1}{2\tau_i\rho} \|\mathbf{y}_{i_t}^{t+1} - \mathbf{y}_{i_t}^{t}\|_2^2 \\
&= \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] + \sum_{i_t \in \mathbb{I}_t} \frac{\tau_{i_t}\rho}{2} \|\mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t}\|_2^2 \,. \qquad (5.107)
\end{aligned}
$$

$\mathbf{x}^{t+1}$ is independent of $\mathbb{I}_t$. Taking expectation over $\mathbb{I}_t$, we have

$$
\frac{K_I}{I} \sum_{i=1}^{I} \langle -\mathbf{y}_i^t, \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \rangle = \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] + \frac{K_I}{I} \sum_{i=1}^{I} \frac{\tau_i\rho}{2} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,. \tag{5.108}
$$

Dividing both sides by $\frac{K_I}{I}$ yields

$$
\langle -\mathbf{y}^t, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle = \frac{I}{K_I} \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] + \sum_{i=1}^{I} \frac{\tau_i\rho}{2} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,. \tag{5.109}
$$

Using (5.7) and the fact that $\mathbf{y}^{t+1} = [\mathbf{y}_{i_t \in \mathbb{I}_t}^{t+1}, \mathbf{y}_{k \notin \mathbb{I}_t}^{t}]^T$, we have

$$
\langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle = \sum_{i_t \in \mathbb{I}_t} \langle \mathbf{y}_{i_t}^t - \mathbf{y}_{i_t}^{t+1}, \mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t} \rangle = -\sum_{i_t \in \mathbb{I}_t} \tau_{i_t}\rho \|\mathbf{A}_{i_t}^r \mathbf{x}^{t+1} - \mathbf{a}_{i_t}\|_2^2 \,. \tag{5.110}
$$

Taking expectation over $\mathbb{I}_t$, we have

$$
\mathbb{E}_{\mathbb{I}_t} \langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle = -\frac{K_I}{I} \sum_{i=1}^{I} \tau_i\rho \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,. \tag{5.111}
$$

Adding (5.109) and (5.111), we have

$$
\begin{aligned}
&\mathbb{E}_{\mathbb{I}_t} \langle -\mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle = \mathbb{E}_{\mathbb{I}_t} \langle -\mathbf{y}^{t}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle + \mathbb{E}_{\mathbb{I}_t} \langle \mathbf{y}^t - \mathbf{y}^{t+1}, \mathbf{A}\mathbf{x}^{t+1} - \mathbf{a} \rangle \\
&= \frac{I}{K_I} \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \left[\|\mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^{t+1}\|_2^2 \right] + \sum_{i=1}^{I} \frac{\tau_i\rho}{2} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 - \frac{K_I}{I} \sum_{i=1}^{I} \tau_i\rho \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \,,
\end{aligned} \tag{5.112}
$$

which completes the proof. ■

The following theorem establishes the iteration complexity of PDMM in an ergodic sense.

Theorem 16 *Let $(\mathbf{x}_{j_t}^t, \mathbf{y}_{i_t}^t)$ be generated by PDMM (5.5)-(5.7). Let $\bar{\mathbf{x}}^T = \sum_{t=1}^T \mathbf{x}^t$. Setting $\nu_i = 1 - \frac{1}{\tilde{K}_i}$ and $\tau_i = \frac{K}{\tilde{K}_i[(2J-K)\frac{K_I}{I} + K(1-\frac{K_I}{I})]}$, we have*

$$\mathbb{E} f(\bar{\mathbf{x}}^T) - f(\mathbf{x}^*) \tag{5.113}$$

$$\leq \frac{\frac{I}{K_I}\sum_{i=1}^I \frac{1}{2\tau_i\rho}\|\mathbf{y}_i^0\|_2^2 + \frac{J}{K}\left\{\frac{1}{2\beta_i\rho}\|\mathbf{y}_i^*\|_2^2 + \tilde{\mathcal{L}}_\rho(\mathbf{x}^1, \mathbf{y}^1) + \frac{\rho}{2}\|\mathbf{z}^* - \mathbf{z}^1\|_{\mathbf{Q}}^2 + \boldsymbol{\eta}^T B_\phi(\mathbf{x}^*, \mathbf{x}^1)\right\}}{T},$$

$$\mathbb{E}\sum_{i=1}^I \beta_i \|\mathbf{A}_i^r \bar{\mathbf{x}}^T - \mathbf{a}_i\|_2^2 \leq \frac{\frac{2}{\rho}h(\mathbf{v}^*, \mathbf{v}^0)}{T}. \tag{5.114}$$

where $\beta_i = \frac{K}{J\tilde{K}_i}$.

Proof: Plugging (5.105) into (5.68) yields

$$\begin{aligned}
f(\mathbf{x}^t) - f(\mathbf{x}^*) &\leq \frac{I}{K_I}\sum_{i=1}^I \frac{1}{2\tau_i\rho}(\|\mathbf{y}_i^{t-1}\|_2^2 - \|\mathbf{y}_i^t\|_2^2) \\
&+ \frac{J}{K}\Bigg\{\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\mathbb{I}_t}\tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2}\sum_{i=1}^I \beta_i \mathbb{E}_{\mathbb{I}_t}\|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \\
&+ \frac{\rho}{2}(\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\mathbb{I}_t}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2) \\
&+ \boldsymbol{\eta}^T(B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\mathbb{I}_t}B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\mathbb{I}_t}B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t))\Bigg\}.
\end{aligned} \tag{5.115}$$

Taking expectaion over ξ_{t-1}, we have

$$\begin{aligned}
\mathbb{E}_{\xi_{t-1}} f(\mathbf{x}^t) - f(\mathbf{x}^*) &\leq \frac{I}{K_I}\sum_{i=1}^I \frac{1}{2\tau_i\rho}(\mathbb{E}_{\xi_{t-2}}\|\mathbf{y}_i^{t-1}\|_2^2 - \mathbb{E}_{\xi_{t-1}}\|\mathbf{y}_i^t\|_2^2) \\
&+ \frac{J}{K}\Bigg\{\mathbb{E}_{\xi_{t-1}}\tilde{\mathcal{L}}_\rho(\mathbf{x}^t, \mathbf{y}^t) - \mathbb{E}_{\xi_t}\tilde{\mathcal{L}}_\rho(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) - \frac{\rho}{2}\sum_{i=1}^I \beta_i \mathbb{E}_{\xi_t}\|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \\
&+ \frac{\rho}{2}(\mathbb{E}_{\xi_{t-1}}\|\mathbf{z}^* - \mathbf{z}^t\|_{\mathbf{Q}}^2 - \mathbb{E}_{\xi_t}\|\mathbf{z}^* - \mathbf{z}^{t+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\xi_t}\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{P}_t}^2) \\
&+ \boldsymbol{\eta}^T(\mathbb{E}_{\xi_{t-1}}B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^t) - \mathbb{E}_{\xi_t}B_{\boldsymbol{\phi}}(\mathbf{x}^*, \mathbf{x}^{t+1}) - \mathbb{E}_{\xi_t}B_{\boldsymbol{\phi}}(\mathbf{x}^{t+1}, \mathbf{x}^t))\Bigg\}.
\end{aligned} \tag{5.116}$$

Summing over t, we have

$$\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_{\xi_{t-1}} f(\mathbf{x}^t) - f(\mathbf{x}^*) \leq \frac{I}{K_I} \sum_{i=1}^{I} \frac{1}{2\tau_i \rho} (\|\mathbf{y}_i^0\|_2^2 - \mathbb{E}_{\xi_{T-1}} \|\mathbf{y}_i^T\|_2^2) \\
&+ \frac{J}{K} \left\{ \tilde{\mathcal{L}}_\rho(\mathbf{x}^1, \mathbf{y}^1) - \mathbb{E}_{\xi_T} \tilde{\mathcal{L}}_\rho(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}) - \frac{\rho}{2} \sum_{t=1}^{T} \sum_{i=1}^{I} \beta_i \mathbb{E}_{\xi_t} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \right. \\
&+ \frac{\rho}{2} (\|\mathbf{z}^* - \mathbf{z}^1\|_{\mathbf{Q}}^2 - \mathbb{E}_{\xi_T} \|\mathbf{z}^* - \mathbf{z}^{T+1}\|_{\mathbf{Q}}^2 - \mathbb{E}_{\xi_T} \|\mathbf{z}^{T+1} - \mathbf{z}^T\|_{\mathbf{Q}}^2) \\
&\left. + \boldsymbol{\eta}^T (B_\phi(\mathbf{x}^*, \mathbf{x}^1) - \mathbb{E}_{\xi_T} B_\phi(\mathbf{x}^*, \mathbf{x}^{T+1}) - \mathbb{E}_{\xi_T} B_\phi(\mathbf{x}^{T+1}, \mathbf{x}^T)) \right\} . \qquad (5.117)
\end{aligned}$$

Following (5.94), we have

$$\begin{aligned}
&\tilde{\mathcal{L}}_\rho(\mathbf{x}^{T+1}, \mathbf{y}^{T+1}) \\
&= f(\mathbf{x}^{T+1}) - f(\mathbf{x}^*) + \sum_{i=1}^{I} [\langle \mathbf{y}_i^{T+1}, \mathbf{A}_i \mathbf{x}^{T+1} - \mathbf{a}_i \rangle + \frac{(\gamma_i - \frac{K_I}{I}\tau_i)\rho}{2} \|\mathbf{A}_i \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2] \\
&\geq - \sum_{i=1}^{I} \langle \mathbf{y}_i^*, \mathbf{A}_i^r \mathbf{x}^{T+1} - \mathbf{a}_i \rangle + \sum_{i=1}^{I} [\langle \mathbf{y}_i^T, \mathbf{A}_i \mathbf{x}^{T+1} - \mathbf{a}_i \rangle + \frac{(\gamma_i + \frac{K_I}{I}\tau_i)\rho}{2} \|\mathbf{A}_i \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2] \\
&\geq - \sum_{i=1}^{I} (\frac{1}{2\delta_i} \|\mathbf{y}_i^*\|_2^2 + \frac{\delta_i}{2} \|\mathbf{A}_i^r \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2) \\
&+ \sum_{i=1}^{I} \left[- \frac{KI}{2JK_I\tau_i\rho} \|\mathbf{y}_i^T\|_2^2 + [\gamma_i + (1 - \frac{J}{K}) \frac{K_I}{I} \tau_i] \frac{\rho}{2} \|\mathbf{A}_i \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2 \right] \\
&\geq - \sum_{i=1}^{I} (\frac{1}{2\delta_i} \|\mathbf{y}_i^*\|_2^2 + \frac{\delta_i}{2} \|\mathbf{A}_i^r \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2) - \sum_{i=1}^{I} \frac{KI}{2JK_I\tau_i\rho} \|\mathbf{y}_i^T\|_2^2 \, , \qquad (5.118)
\end{aligned}$$

where $\delta_i > 0$ and the last inequality uses (5.96).

Plugging into (5.117), we have

$$\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_{\xi_{t-1}} f(\mathbf{x}^t) - f(\mathbf{x}^*) \\
&\leq \frac{I}{K_I} \sum_{i=1}^{I} \frac{1}{2\tau_i\rho} \|\mathbf{y}_i^0\|_2^2 + \frac{J}{K} \left\{ \tilde{\mathcal{L}}_\rho(\mathbf{x}^1, \mathbf{y}^1) + \frac{\rho}{2} \|\mathbf{z}^* - \mathbf{z}^1\|_{\mathbf{Q}}^2 + \boldsymbol{\eta}^T B_\phi(\mathbf{x}^*, \mathbf{x}^1) \right\} \\
&+ \frac{J}{K} \left\{ \sum_{i=1}^{I} \left[\frac{1}{2\delta_i} \|\mathbf{y}_i^*\|_2^2 + \frac{\delta_i - \beta_i\rho}{2} \mathbb{E} \|\mathbf{A}_i^r \mathbf{x}^{T+1} - \mathbf{a}_i\|_2^2 \right] \right\} . \qquad (5.119)
\end{aligned}$$

Settin $\delta_i = \beta_i \rho$, dividing by T and letting $\bar{\mathbf{x}}^T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}^t$ complete the proof.

Dividing both sides of (5.104) by T yields (5.114). ■

5.B Connection to ADMM

We use ADMM to solve (5.1), similar as [207, 153] but with different forms. We show that ADMM is a speical case of PDMM. The connection can help us understand why the two parameters τ_i, ν_i in PDMM are necessary. We first introduce splitting variables $\mathbf{z}_i$ as follows:

$$\min \sum_{j=1}^{J} f_j(\mathbf{x}_j) \quad \text{s.t.} \quad \mathbf{A}_j \mathbf{x}_j = \mathbf{z}_j, \sum_{j=1}^{J} \mathbf{z}_j = \mathbf{a} \,, \tag{5.120}$$

which can be written as

$$\min \sum_{j=1}^{K} f_j(\mathbf{x}_j) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}_j \mathbf{x}_j = \mathbf{z}_j \,, \tag{5.121}$$

where $g(\mathbf{z})$ is an indicator function of $\sum_{j=1}^{K} \mathbf{z}_j = \mathbf{a}$. The augmented Lagrangian is

$$\mathcal{L}_\rho(\mathbf{x}_j, \mathbf{z}_j, \mathbf{y}_j) = \sum_{j=1}^{J} \left[f_j(\mathbf{x}_j) + \langle \mathbf{y}_j, \mathbf{A}_j \mathbf{x}_j - \mathbf{z}_j \rangle + \frac{\rho}{2} \| \mathbf{A}_j \mathbf{x}_j - \mathbf{z}_j \|_2^2 \right] \,, \tag{5.122}$$

where $\mathbf{y}_j$ is the dual variable. We have the following ADMM iterates:

$$\mathbf{x}_j^{t+1} = \text{argmin}_{\mathbf{x}_i} \ f_j(\mathbf{x}_j) + \langle \mathbf{y}_j^t, \mathbf{A}_j \mathbf{x}_j - \mathbf{z}_j^t \rangle + \frac{\rho}{2} \| \mathbf{A}_j \mathbf{x}_j - \mathbf{z}_j^t \|_2^2 \,, \tag{5.123}$$

$$\mathbf{z}^{t+1} = \text{argmin}_{\sum_{j=1}^{K} \mathbf{z}_j = \mathbf{a}} \sum_{j=1}^{K} \left[\langle \mathbf{y}_i^t, \mathbf{A}_j \mathbf{x}_j^{t+1} - \mathbf{z}_j \rangle + \frac{\rho}{2} \| \mathbf{A}_j \mathbf{x}_j^{t+1} - \mathbf{z}_j \|_2^2 \right] \,, \tag{5.124}$$

$$\mathbf{y}_j^{t+1} = \mathbf{y}_j^t + \rho(\mathbf{A}_j \mathbf{x}_j^{t+1} - \mathbf{z}_j^{t+1}) \,. \tag{5.125}$$

The Lagrangian of (5.124) is

$$\mathcal{L} = \sum_{j=1}^{J} \left[\langle \mathbf{y}_j^t, \mathbf{A}_j \mathbf{x}_j^{t+1} - \mathbf{z}_j \rangle + \frac{\rho}{2} \| \mathbf{A}_j \mathbf{x}_j^{t+1} - \mathbf{z}_j \|_2^2 \right] + \langle \boldsymbol{\lambda}, \sum_{j=1}^{J} \mathbf{z}_j - \mathbf{a} \rangle \,, \tag{5.126}$$

where $\boldsymbol{\lambda}$ is the dual variable. The first order optimality is

$$-\mathbf{y}_j^t + \rho(\mathbf{z}_j^{t+1} - \mathbf{A}_j \mathbf{x}_j^{t+1}) + \boldsymbol{\lambda} = 0 \,. \tag{5.127}$$

Using (5.125) gives

$$\boldsymbol{\lambda} = \mathbf{y}_j^{t+1}, \quad \forall j \; . \tag{5.128}$$

Denoting $\mathbf{y}^t = \mathbf{y}_j^t$, (5.127) becomes

$$\mathbf{y}^{t+1} = \mathbf{y}^t + \rho(\mathbf{A}_j\mathbf{x}_j^{t+1} - \mathbf{z}_j^{t+1}) \; . \tag{5.129}$$

Summing over j and using the constraint $\sum_{j=1}^J \mathbf{z}_i = \mathbf{a}$, we have

$$\mathbf{y}^{t+1} = \mathbf{y}^t + \frac{\rho}{J}(\mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}) \; . \tag{5.130}$$

Subtracting (5.129) from (5.130), simple calculations yields

$$\mathbf{z}_j^{t+1} = \mathbf{A}_j\mathbf{x}_j^{t+1} + \frac{1}{J}(\mathbf{A}\mathbf{x}^{t+1} - \mathbf{a}) \; . \tag{5.131}$$

Plugging back int (5.123), we have

$$\begin{aligned}
\mathbf{x}_j^{t+1} &= \text{argmin}_{\mathbf{x}_j} \;\; f_j(\mathbf{x}_j) + \langle \mathbf{y}^t, \mathbf{A}_j\mathbf{x}_j \rangle + \frac{\rho}{2}\|\mathbf{A}_j\mathbf{x}_j - \mathbf{z}_j^t\|_2^2 \\
&= \text{argmin}_{\mathbf{x}_j} \;\; f_j(\mathbf{x}_j) + \langle \mathbf{y}^t, \mathbf{A}_j\mathbf{x}_j \rangle + \frac{\rho}{2}\|\mathbf{A}_j\mathbf{x}_j - \mathbf{A}_j\mathbf{x}_j^t + \frac{\mathbf{A}\mathbf{x}^t - \mathbf{a}}{J}\|_2^2 \\
&= \text{argmin}_{\mathbf{x}_j} \;\; f_j(\mathbf{x}_j) + \langle \hat{\mathbf{y}}^t, \mathbf{A}_j\mathbf{x}_j \rangle + \frac{\rho}{2}\|\mathbf{A}_j\mathbf{x}_j + \sum_{k\neq j}\mathbf{A}_k\mathbf{x}_k^t - \mathbf{a}\|_2^2 \; ,
\end{aligned} \tag{5.132}$$

where $\hat{\mathbf{y}}^t = \mathbf{y}^t - (1 - \frac{1}{J})\rho(\mathbf{A}\mathbf{x}^t - \mathbf{a})$, which becomes PDMM by setting $\tau = \frac{1}{J}, \nu = 1 - \frac{1}{J}$ and updating all blocks. Therefore, sADMM is a special case of PDMM.

5.C Connection to PJADMM

We consider the case when all blocks are used in PDMM. We show that if setting η_j sufficiently large, the dual backward step (5.5) is not needed, which becomes PJADMM [47].

Corollary 2 *Let $\{\mathbf{x}_j^t, \mathbf{y}_i^t\}$ be generated by PDMM (5.6)-(5.5). Assume $\tau_i > 0$ and $\nu_i \geq 0$. We have*

$$\begin{aligned}
f(\mathbf{x}^{t+1}) - f(\mathbf{x}^*) &\leq \sum_{i=1}^I \left\{ -\langle \mathbf{y}_i^{t+1}, \mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i \rangle + \frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \right\} \\
&+ \frac{\rho}{2}(\|\mathbf{z}^t - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{Q}}^2)
\end{aligned}$$

$$+\frac{\rho}{2}\sum_{i=1}^{I}\Big\{(\nu_i-1+\frac{1}{d_i})(\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2)$$
$$+(\tau_i+2\nu_i-2)\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2+(1-\nu_i-\frac{1}{d_i})\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2\Big\}$$
$$+\sum_{j=1}^{J}\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})-B_{\phi_j}(\mathbf{x}_j^{t+1},\mathbf{x}_j^t)\right)\ . \tag{5.133}$$

Proof: Let $\mathbb{I}_t$ be all blocks, $K=J$. According the definition of $\mathbf{P}_t$ in (5.33) and $\mathbf{Q}$ in (5.39), $\mathbf{P}_t=\mathbf{Q}$. Therefore, (5.42) reduces to

$$f(\mathbf{x}^{t+1})-f(\mathbf{x}^*)\le\sum_{i=1}^{I}\Big\{-\langle\mathbf{y}_i^{t+1},\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\rangle+\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\Big\}$$
$$+\frac{\rho}{2}(\|\mathbf{z}^t-\mathbf{z}^*\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^*\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{Q}}^2)$$
$$+\sum_{j=1}^{J}\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})-B_{\phi_j}(\mathbf{x}_j^{t+1},\mathbf{x}_j^t)\right)$$
$$+\frac{\rho}{2}\sum_{i=1}^{I}\Big\{(\nu_i-1+\frac{1}{d_i})\|\mathbf{A}_i^r\mathbf{x}^t-\mathbf{a}_i\|_2^2-(1-\nu_i-\tau_i+\frac{1}{d_i})\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2$$
$$+(1-\nu_i-\frac{1}{d_i})\|\mathbf{A}_i^r(\mathbf{x}^{t+1}-\mathbf{x}^t)\|_2^2\Big\}\ . \tag{5.134}$$

Rearranging the terms completes the proof. ■

Corollary 3 *Let $\{\mathbf{x}_j^t,\mathbf{y}_i^t\}$ be generated by PDMM (5.6)-(5.5). Assume (1)$\tau_i>0$ and $\nu_i\ge 0$; (2) $\eta_j>0$; (3) ϕ_j is α_j-strongly convex. We have*

$$f(\mathbf{x}^{t+1})-f(\mathbf{x}^*)\le\sum_{i=1}^{I}\Big\{-\langle\mathbf{y}_i^{t+1},\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\rangle+\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\Big\}$$
$$+\frac{\rho}{2}(\|\mathbf{z}^t-\mathbf{z}^*\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^*\|_{\mathbf{Q}}^2-\|\mathbf{z}^{t+1}-\mathbf{z}^t\|_{\mathbf{Q}}^2)$$
$$+\sum_{j=1}^{J}\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})\right)\ . \tag{5.135}$$

ν_i and τ_i satisfy $\nu_i\in[1-\frac{1}{d_i}-\frac{\eta_j\alpha_j}{\rho I d_i\lambda_{\max}^{ij}},1-\frac{1}{d_i}]$ and $\tau_i\le 1+\frac{1}{d_i}-\nu_i$, where $\lambda_{\max}^{ij}$ is the largest eigenvalue of $\mathbf{A}_{ij}^T\mathbf{A}_{ij}$. In particular, if $\eta_j=\frac{(d_i-1)\rho I\lambda_{\max}^{ij}}{\alpha_j}$, $\nu_i=0$ and $\tau_i\le 1+\frac{1}{d_i}$.

Proof: Assume $\eta_j > 0$. We can choose larger τ_i and smaller ν_i than Lemma 16 by setting η_j sufficiently large. Since ϕ_j is α_j-strongly convex, $B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) \geq \frac{\alpha_j}{2}\|\mathbf{x}_j^{t+1} - \mathbf{x}_j^t\|_2^2$. We have

$$\sum_{j=1}^{J} \eta_j B_{\phi_j}(\mathbf{x}_j^{t+1}, \mathbf{x}_j^t) \geq \sum_{i=1}^{I}\sum_{j=1}^{J} \frac{\eta_j \alpha_j}{2I}\|\mathbf{x}_j^{t+1} - \mathbf{x}_j^t\|_2^2 \geq \sum_{i=1}^{I}\sum_{j\in\mathcal{N}(i)} \frac{\eta_j \alpha_j}{2I\lambda_{\max}^{ij}}\|\mathbf{A}_{ij}(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)\|_2^2 \,. \tag{5.136}$$

$$\|\mathbf{A}_i^r(\mathbf{x}^{t+1} - \mathbf{x}^t)\|_2^2 = \|\sum_{j\in\mathcal{N}(i)} \mathbf{A}_{ij}(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)\|_2^2 \leq d_i \sum_{j\in\mathcal{N}(i)} \|\mathbf{A}_{ij}(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)\|_2^2 \,, \tag{5.137}$$

where $\lambda_{\max}^{ij}$ is the largest eigenvalue of $\mathbf{A}_{ij}^T\mathbf{A}_{ij}$. Plugging into (5.133) gives

$$\begin{aligned}
f(\mathbf{x}^{t+1}) - f(\mathbf{x}^*) &\leq \sum_{i=1}^{I}\left\{-\langle \mathbf{y}_i^{t+1}, \mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\rangle + \frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\right\} \\
&+ \frac{\rho}{2}(\|\mathbf{z}^t - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{Q}}^2) \\
&+ \frac{\rho}{2}\sum_{i=1}^{I}\left\{(\nu_i - 1 + \frac{1}{d_i})(\|\mathbf{A}_i^r\mathbf{x}^t - \mathbf{a}_i\|_2^2 - \|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2)\right. \\
&\left. +(\tau_i + 2\nu_i - 2)\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 + \sum_{j\in\mathcal{N}(i)}[(1-\nu_i)d_i - 1 - \frac{\eta_j\alpha_j}{\rho I\lambda_{\max}^{ij}}]\|\mathbf{A}_{ij}(\mathbf{x}_j^{t+1} - \mathbf{x}_j^t)\|_2^2\right\} \\
&+ \sum_{j=1}^{J}\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) - B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^{t+1})\right) \,.
\end{aligned} \tag{5.138}$$

If $(1-\nu_i)d_i - 1 - \frac{\eta_j\alpha_j}{\rho I\lambda_{\max}^{ij}} \leq 0$, i.e., $\nu_i \geq 1 - \frac{1}{d_i} - \frac{\eta_j\alpha_j}{\rho I d_i\lambda_{\max}^{ij}}$, we have

$$\begin{aligned}
f(\mathbf{x}^{t+1}) - f(\mathbf{x}^*) &\leq \frac{\rho}{2}\sum_{i=1}^{I}\left\{-\frac{2}{\rho}\langle \mathbf{y}_i^{t+1}, \mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\rangle + \tau_i\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\right\} \\
&+ \frac{\rho}{2}(\|\mathbf{z}^t - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^*\|_{\mathbf{Q}}^2 - \|\mathbf{z}^{t+1} - \mathbf{z}^t\|_{\mathbf{Q}}^2) \\
&+ \sum_{i=1}^{J}\eta_i\left(B_{\phi_i}(\mathbf{x}_j^*, \mathbf{x}_j^t) - B_{\phi_i}(\mathbf{x}_j^*, \mathbf{x}_j^{t+1})\right) \\
&+ \frac{\rho}{2}\sum_{i=1}^{I}\left\{-(\nu_i - 1 + \frac{1}{d_i})\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 + (\tau_i - 2 + 2\nu_i)\|\mathbf{A}_i^r\mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2\right\} \,.
\end{aligned} \tag{5.139}$$

If $\tau_i - 2 + 2\nu_i - (\nu_i - 1 + \frac{1}{d_i}) \le 0$, i.e., $\tau_i \le 1 + \frac{1}{d_i} - \nu_i$, the last two terms in (5.139) can be removed. Therefore, when $\nu_i \ge 1 - \frac{1}{d_i} - \frac{\eta_j \alpha_j}{\rho I d_i \lambda_{\max}^{ij}}$ and $\tau_i \le 1 + \frac{1}{d_i} - \nu_i$, we have (5.135). ■

Define the current iterate $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ and $h(\mathbf{v}^*, \mathbf{v}^t)$ as a distance from $\mathbf{v}^t$ to a KKT point $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$:

$$h(\mathbf{v}^*, \mathbf{v}^t) = \sum_{i=1}^{I} \frac{1}{2\tau_i \rho} \|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 + \frac{\rho}{2}\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{Q}}^2 + \sum_{j=1}^{J} \eta_j B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) \ . \tag{5.140}$$

The following theorem shows that $h(\mathbf{v}^*, \mathbf{v}^t)$ decreases monotonically and thus establishes the global convergence of PDMM.

Theorem 17 ***(Global Convergence of PDMM)*** *Let $\mathbf{v}^t = (\mathbf{x}_j^t, \mathbf{y}_i^t)$ be generated by PDMM (5.6)-(5.5) and $\mathbf{v}^* = (\mathbf{x}_j^*, \mathbf{y}_i^*)$ be a KKT point satisfying (5.82)-(5.83). Assume τ_i, ν_i and γ_i satisfy conditions in Lemma 3. Then $\mathbf{v}^t$ converges to the KKT point $\mathbf{v}^*$ monotonically, i.e.,*

$$h(\mathbf{v}^*, \mathbf{v}^{t+1}) \le h(\mathbf{v}^*, \mathbf{v}^t) \tag{5.141}$$

Proof: Adding (5.94) and (5.135) together yields

$$\begin{aligned} 0 \le & \sum_{i=1}^{I} \left\{ \langle \mathbf{y}_i^* - \mathbf{y}_i^{t+1}, \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \rangle + \frac{\tau_i \rho}{2} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \right\} \\ & + \frac{\rho}{2}(\|\mathbf{u}^t - \mathbf{u}^*\|_{\mathbf{Q}}^2 - \|\mathbf{u}^{t+1} - \mathbf{u}^*\|_{\mathbf{Q}}^2 - \|\mathbf{u}^{t+1} - \mathbf{u}^t\|_{\mathbf{Q}}^2) \\ & + \sum_{j=1}^{J} \eta_j \left(B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^t) - B_{\phi_j}(\mathbf{x}_j^*, \mathbf{x}_j^{t+1}) \right) \ . \end{aligned} \tag{5.142}$$

The first term in the bracket can be rewritten as

$$\begin{aligned} \langle \mathbf{y}_i^* - \mathbf{y}_i^{t+1}, \mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i \rangle &= \frac{1}{\tau_i \rho} \langle \mathbf{y}_i^* - \mathbf{y}_i^{t+1}, \mathbf{y}_i^{t+1} - \mathbf{y}_i^t \rangle \\ &= \frac{1}{2\tau_i \rho} \left(\|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^{t+1}\|_2^2 - \|\mathbf{y}_i^{t+1} - \mathbf{y}_i^t\|_2^2 \right) \\ &= \frac{1}{2\tau_i \rho} \left(\|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^{t+1}\|_2^2 \right) - \frac{\tau_i \rho}{2} \|\mathbf{A}_i^r \mathbf{x}^{t+1} - \mathbf{a}_i\|_2^2 \ . \end{aligned} \tag{5.143}$$

Plugging back into (5.142) yields

$$0 \le \sum_{i=1}^{I} \frac{1}{2\tau_i \rho} \left(\|\mathbf{y}_i^* - \mathbf{y}_i^t\|_2^2 - \|\mathbf{y}_i^* - \mathbf{y}_i^{t+1}\|_2^2 \right)$$

$$+\frac{\rho}{2}(\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{Q}}^2-\|\mathbf{u}^{t+1}-\mathbf{u}^*\|_{\mathbf{Q}}^2-\|\mathbf{u}^{t+1}-\mathbf{u}^t\|_{\mathbf{Q}}^2)$$
$$+\sum_{j=1}^J\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})\right)\ . \tag{5.144}$$

Rearranging the terms completes the proof. ■

The following theorem establishes the $O(1/T)$ convergence rate for the objective in an ergodic sense.

Theorem 18 *Let $(\mathbf{x}_j^t,\mathbf{y}_i^t)$ be generated by PDMM (5.6)-(5.5). Assume $\tau_i,\nu_i\geq 0$ satisfy conditions in Lemma 3. Let $\bar{\mathbf{x}}^T=\sum_{t=1}^T\mathbf{x}^t$. We have*

$$f(\bar{\mathbf{x}}^T)-f(\mathbf{x}^*)\leq\frac{\frac{1}{2\tau\rho}\|\mathbf{y}^0\|_2^2+\frac{\rho}{2}\|\mathbf{u}^0-\mathbf{u}^*\|_{\mathbf{Q}}^2+\sum_{j=1}^J\eta_jB_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^0)}{T}\ , \tag{5.145}$$

Proof: Using (5.7), we have

$$-\langle\mathbf{y}_i^{t+1},\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\rangle=-\frac{1}{\tau_i\rho}\langle\mathbf{y}_i^{t+1},\mathbf{y}_i^{t+1}-\mathbf{y}_i^t\rangle$$
$$=\frac{1}{2\tau_i\rho}(\|\mathbf{y}_i^t\|_2^2-\|\mathbf{y}_i^{t+1}\|_2^2-\|\mathbf{y}_i^{t+1}-\mathbf{y}_i^t\|_2^2)$$
$$=\frac{1}{2\tau_i\rho}(\|\mathbf{y}_i^t\|_2^2-\|\mathbf{y}_i^{t+1}\|_2^2)-\frac{\tau_i\rho}{2}\|\mathbf{A}_i^r\mathbf{x}^{t+1}-\mathbf{a}_i\|_2^2\ . \tag{5.146}$$

Plugging into (5.135) yields

$$f(\mathbf{x}^{t+1})-f(\mathbf{x}^*)\leq\sum_{i=1}^I\frac{1}{2\tau_i\rho}(\|\mathbf{y}_i^t\|_2^2-\|\mathbf{y}_i^{t+1}\|_2^2)$$
$$+\frac{\rho}{2}(\|\mathbf{u}^t-\mathbf{u}^*\|_{\mathbf{Q}}^2-\|\mathbf{u}^{t+1}-\mathbf{u}^*\|_{\mathbf{Q}}^2-\|\mathbf{u}^{t+1}-\mathbf{u}^t\|_{\mathbf{Q}}^2)$$
$$+\sum_{j=1}^J\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})\right)\ . \tag{5.147}$$

Summing over t from 0 to $T-1$, we have

$$\sum_{t=0}^{T-1}\left[f(\mathbf{x}^{t+1})-f(\mathbf{x}^*)\right]\leq\sum_{i=1}^I\frac{1}{2\tau_i\rho}(\|\mathbf{y}_i^t\|_2^2-\|\mathbf{y}_i^{t+1}\|_2^2)$$
$$+\frac{\rho}{2}(\|\mathbf{u}^0-\mathbf{u}^*\|_{\mathbf{Q}}^2-\|\mathbf{u}^T-\mathbf{u}^*\|_{\mathbf{Q}}^2)$$
$$+\sum_{j=1}^J\eta_j\left(B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^t)-B_{\phi_j}(\mathbf{x}_j^*,\mathbf{x}_j^{t+1})\right)\ . \tag{5.148}$$

Applying the Jensen's inequality on the LHS and using $\bar{\mathbf{x}}^T = \sum_{t=1}^T \mathbf{x}^t$ complete the proof. ■

If $\eta_j = \frac{(d_i-1)\rho I \lambda_{\max}^{ij}}{\alpha_j}$, $\nu_i = 0$ and $\tau_i = 1$. Therefore, PDMM becomes PJADMM [47], where the convergence rate of PJADMM has been improved to $o(1/T)$.

Chapter 6

Online Alternating Direction Method of Multipliers

6.1 Introduction

In recent years, online optimization [27, 225, 75] and its batch counterpart stochastic gradient descent [162, 95] has contributed substantially to advances in large scale optimization techniques for machine learning. Online convex optimization has been generalized to handle time-varying and non-smooth convex functions [52, 53, 210]. Distributed optimization, where the problem is divided into parts on which progress can be made in parallel, has also contributed to advances in large scale optimization [19, 15, 25].

Important advances have been made based on the above ideas in the recent literature. Composite objective mirror descent (COMID) [52] generalizes mirror descent [10] to the online setting. COMID also includes certain other proximal splitting methods such as FOBOS [53] as special cases. Regularized dual averaging (RDA) [210] generalizes dual averaging [147] to online and composite optimization, and can be used for distributed optimization [50]. The three methods consider the following composite objective optimization [146]:

$$\min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} (f_t(\mathbf{x}) + g(\mathbf{x})) \; , \tag{6.1}$$

where the functions f_t, g are convex functions and $\mathcal{X}$ is a convex set. Solving (6.1) usually involves the projection onto $\mathcal{X}$. In some cases, e.g., when g is the ℓ_1 norm or $\mathcal{X}$ is the unit simplex, the projection can be done efficiently. In general, the full projection requires an inner loop algorithm, leading to a double loop algorithm for solving (6.1) [77].

In this chapter, we propose single loop online optimization algorithms for composite objective optimization subject to linear constraints. In particular, we consider optimization problems of the following form:

$$\min_{\mathbf{x}\in\mathcal{X},\mathbf{z}\in\mathcal{Z}} \sum_{t=1}^{T}(f_t(\mathbf{x})+g(\mathbf{z})) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}\ , \tag{6.2}$$

where $\mathbf{A}\in\mathbb{R}^{m\times n_1}, \mathbf{B}\in\mathbb{R}^{m\times n_2}, \mathbf{c}\in\mathbb{R}^m$, $\mathbf{x}\in\mathcal{X}\in\mathbb{R}^{n_1\times 1}, \mathbf{z}\in\mathcal{Z}\in\mathbb{R}^{n_2\times 1}$ and $\mathcal{X}$ and $\mathcal{Z}$ are convex sets. The linear equality constraint introduces splitting variables and thus splits functions and feasible sets into simpler constraint sets $\mathbf{x}\in\mathcal{X}$ and $\mathbf{z}\in\mathcal{Z}$. (6.2) can easily accommodate linear inequality constraints by introducing a slack variable, which will be discussed in Section 6.5.4. In the sequel, we drop the convex sets $\mathcal{X}$ and $\mathcal{Z}$ for ease of exposition, noting that one can consider g and other additive functions to be the indicators of suitable convex feasible sets. f_t and g can be non-smooth, including piecewise linear and indicator functions. In the context of machine learning, f_t is usually a loss function such as ℓ_1, ℓ_2, hinge and logistic loss, while g is a regularizer, e.g., ℓ_1, ℓ_2, nuclear norm, mixed-norm and total variation.

We consider two scenarios in the online setting, based on whether an additional Bregman divergence is needed or not for a proximal function in each step. We propose efficient online ADM (OADM) algorithms for both scenarios which make a single pass through the update equations and avoid a double loop algorithm. In the online setting, while a single pass through the ADM update equations is not guaranteed to satisfy the linear constraint $\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}$ in each iteration, we consider two types of regret: regret in the *objective* as well as regret in *constraint violation*. We establish both types of regret bounds for general and strongly convex functions. In Table 6.1, we summarize the main results of OADM and also compare with OGD [225], FOBOS [53], COMID [52] and RDA [210]. While OADM aims to solve linearly-constrained composite objective optimization problems, OGD, FOBOS and RDA are for such problems without explicit constraints. In both general and strongly convex cases, our methods achieve the optimal regret bounds for the objective as well as the constraint violation, while start-of-the-art methods achieve the optimal regret bounds for the objective. We also present preliminary experimental results illustrating the performance of the proposed OADM algorithms in comparison with FOBOS and RDA [53, 210].

Problem	$\min_{\mathbf{Ax}+\mathbf{Bz}=\mathbf{c}} \sum_t f_t(\mathbf{x}) + g(\mathbf{z})$		$\min_{\mathbf{x}} \sum_t f_t(\mathbf{x}) + g(\mathbf{x})$
Methods	OADM		OGD, FOBOS, COMID, RDA
Regret Bounds	Objective	constraint	Objective
General Convex	$O(\sqrt{T})$	$O(\sqrt{T})$	$O(\sqrt{T})$
Strongly Convex	$O(\log(T))$	$O(\log(T))$	$O(\log(T))$

Table 6.1: Main results for regret bounds of OADM in solving linearly-constrained composite objective optimization, in comparison with OGD, FOBOS, COMID and RDA in solving composite objective optimization. In both general and strongly convex cases, OADM achieves the optimal regret bounds for the objective, matching the results of the state-of-the-art methods. In addition, OADM also achieves the optimal regret bounds for constraint violation, showing the equality constraint will be satisfied on average.

The key advantage of the OADM algorithms can be summarized as follows: Like COMID and RDA, OADM can solve online composite optimization problems, matching the regret bounds for existing methods. The ability to additionally handle linear equality constraint of the form $\mathbf{Ax} + \mathbf{Bz} = \mathbf{c}$ makes non-trivial variable splitting possible yielding efficient distributed online optimization algorithms [44] and projection-free online learning [77] based on OADM. Further, the notion of regret in both the objective as well as constraint may contribute towards development of suitable analysis tools for online constrained optimization problems [131, 128].

The rest of the chapter is organized as follows. In Section 6.2, we propose OADM to solve the online optimization problem with linear constraints. In Section 6.3 and 6.4, we present the regret analysis in two different scenarios based on whether an additional Bregman divergence is added or not. In Section 6.5, we discuss inexact ADM updates and show the stochastic convergence rates, show the connection to related works and projection-free online learning based on OADM. We present preliminary experimental results in Section 6.6.

6.2 Online Alternating Direction Method

In this section, we extend ADM to the online learning setting. Specifically, we focus on using online ADM (OADM) to solve the problem (6.2). For our analysis, $\mathbf{A}$ and $\mathbf{B}$ are assumed to be fixed. At round t, we consider solving the following regularized optimization problem:

$$\mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}} f_t(\mathbf{x}) + g(\mathbf{z}) + \eta B_\phi(\mathbf{x}, \mathbf{x}_t) \ , \tag{6.3}$$

where $\eta \geq 0$ is a learning rate and $B_\phi(\mathbf{x}, \mathbf{x}_t)$ is a Bregman divergence [5, 25].

Let $\phi : \Omega \to \mathbb{R}$ be a continuously differentiable and strictly convex function. Denote $\nabla\phi(\mathbf{y})$ as the gradient of ϕ at $\mathbf{y}$. The Bregman divergence $B_\phi : \Omega \times \mathrm{ri}(\Omega) \to \mathbb{R}_+$ is defined as

$$B_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \nabla\phi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \ .$$

Two widely used examples are squared Euclidian distance $B_\phi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2$ and KL divergence $B_\phi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n x_i \log \frac{x_i}{y_i}$.

If the problem (6.3) is solved exactly in every step, standard analysis techniques [75] can be suitably adopted to obtain sublinear regret bounds. While (6.3) can be solved by batch ADM, we essentially obtain a double loop algorithm where the function f_t changes in the outer loop and the inner loop runs ADM iteratively till convergence so that the constraint are satisfied. Note that existing online methods, such as projected gradient descent and variants [75, 52] do assume a black-box approach for projecting onto the feasible set, which for linear constraint may require iterative cyclic projections [25].

For our analysis, instead of requiring the equality constraint to be satisfied at each time t, we only require the equality constraint to be satisfied in the long run, with a notion of regret associated with constraint. In particular, we consider the following constrained cumulative regret for the online learning problem:

$$\begin{aligned} &\sum_{t=1}^T f_t(\mathbf{x}_t) + g(\mathbf{z}_t) - \min_{\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}} \sum_{t=1}^T f_t(\mathbf{x}) + g(\mathbf{z}) \\ \text{s.t.} \quad &\sum_{t=1}^T \|\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 = o(T) \ , \end{aligned} \tag{6.4}$$

where the cumulative constraint violation is sublinear in T. The goal is to design a single-loop algorithm for (6.4), which has sublinear regret in both the objective and the constraint violation.

The augmented Lagrangian of (6.3) at time t is

$$L_\rho^t(\mathbf{x}, \mathbf{y}, \mathbf{z}) = f_t(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{y}, \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c} \rangle + \eta B_\phi(\mathbf{x}, \mathbf{x}_t) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} - \mathbf{c}\|^2 \ . \tag{6.5}$$

Algorithm 5 Online Alternating Direction Method (OADM)

```
1: Input: $f_t(\mathbf{x}) + g(\mathbf{z}), \mathbf{A}, \mathbf{B}, \mathbf{c}, \rho, \eta, \phi(\mathbf{x})$
2: Initialization: $\mathbf{x}_1, \mathbf{z}_1, \mathbf{u}_1 = \mathbf{0}$
3: for $t = 1$ to $T$ do
4:     $\mathbf{x}_{t+1} = \text{argmin}_{\mathbf{x}}\ f_t(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\|^2 + \eta B_\phi(\mathbf{x}, \mathbf{x}_t)$ ,
5:     $\mathbf{z}_{t+1} = \text{argmin}_{\mathbf{z}}\ g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{Ax}_{t+1} + \mathbf{Bz} - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz} - \mathbf{c}\|^2$ ,
6:     $\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c})$ .
7:     Receive a cost function $f_{t+1}$ and incur loss $f_{t+1}(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1})$ and constraint violation
       $\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2$;
8: end for
```

At time t, OADM (**Algorithm 5**) consists of just one pass through the following three update steps:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\text{argmin}} \left\{ f_t(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{Ax} + \mathbf{Bz}_t - \mathbf{c}\|^2 + \eta B_\phi(\mathbf{x}, \mathbf{x}_t) \right\} , \quad (6.6)$$

$$\mathbf{z}_{t+1} = \underset{\mathbf{z}}{\text{argmin}} \left\{ g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{Ax}_{t+1} + \mathbf{Bz} - \mathbf{c}\rangle + \frac{\rho}{2}\|\mathbf{Ax}_{t+1} + \mathbf{Bz} - \mathbf{c}\|^2 \right\} , \quad (6.7)$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}) . \quad (6.8)$$

Operationally, in round t, the algorithm presents a solution $\{\mathbf{x}_t, \mathbf{z}_t\}$ as well as $\mathbf{y}_t$. Then, nature reveals function f_t and we encounter two types of losses. The first type is the traditional loss measured by $f_t(\mathbf{x}_t) + g(\mathbf{z}_t)$, with corresponding cumulative regret

$$R_1(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) + g(\mathbf{z}_t) - \min_{\mathbf{Ax}+\mathbf{Bz}=\mathbf{c}} \sum_{t=1}^{T} f_t(\mathbf{x}) + g(\mathbf{z}) . \quad (6.9)$$

The second type is the residual of constraint violation, i.e., $\|\mathbf{Ax}_t + \mathbf{Bz}_t - \mathbf{c}\|^2$. As the updates include the primal and dual variables, in line with batch ADM, we use the following cumulative regret for constraint violation:

$$R^c(T) = \sum_{t=1}^{T} \|\mathbf{Ax}_{t+1} + \mathbf{Bz}_{t+1} - \mathbf{c}\|_2^2 + \|\mathbf{Bz}_{t+1} - \mathbf{Bz}_t\|_2^2 . \quad (6.10)$$

The goal is to establish sublinear regret bounds for both the objective and constraint violation.

The OADM updates (6.6)-(6.7) are similar as ADM updates (3.3)-(3.4) except the $\mathbf{x}$ update in OADM uses a time varying function f_t and an additional Bregman divergence, which is the

Regret bounds	$\eta > 0$		$\eta = 0$	
	R_1	R^c	R_2	R^c
general convex	$O(\sqrt{T})$	$O(\sqrt{T})$	$O(\sqrt{T})$	$O(\sqrt{T})$
strongly convex	$O(\log T)$	$O(\log T)$	$O(\log T)$	$O(\log T)$

Table 6.2: Regret Bounds for Online Alternating Direction Method

first scenario where the regret bounds of R_1 (6.9) and R^c (6.10) will be presented in Section 4. We also consider another scenario, where $\eta = 0$ in (6.6) and thus the Bregman divergence is eliminated and only the quadratic penalty term is involved in the $\mathbf{x}$-update. $\mathbf{x}_{t+1}$ is kept close to $\mathbf{x}_t$ indirectly through the quadratic penalty term at $\mathbf{z}_t$. Instead of using $\{\mathbf{x}_t, \mathbf{z}_t\}$ as the solution at round t, we use a solution $\{\hat{\mathbf{x}}_t, \mathbf{z}_t\}$ based on $\mathbf{z}_t$ such that $\mathbf{A}\hat{\mathbf{x}}_t + \mathbf{B}\mathbf{z}_t = \mathbf{c}$. While $\{\hat{\mathbf{x}}_t, \mathbf{z}_t\}$ satisfies the constraint by design, the goal is to establish sublinear regret of the objective $f_t(\hat{\mathbf{x}}_t) + g(\mathbf{z}_t)$, i.e.,

$$R_2(T) = \sum_{t=1}^{T} f_t(\hat{\mathbf{x}}_t) + g(\mathbf{z}_t) - \min_{\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}} \sum_{t=1}^{T} f_t(\mathbf{x}) + g(\mathbf{z}) \ . \tag{6.11}$$

The sublinear regret of constraint violation for the true $\{\mathbf{x}_t, \mathbf{z}_t\}$ defined in (6.10) should still be achieved. The regret bounds for OADM in the two scenarios are summarized in Table 6.2.

Before getting into the regret analysis, we discuss some example problems which can be solved using OADM. Like FOBOS and RDA, OADM can deal with machine learning problems where f_t is a loss function and g is a regularizer, e.g., generalized lasso and group lasso [19, 189, 210] using ℓ_1 or mixed norm, or an indicator function of a convex set. OADM can also be used to solve the batch optimization problems mentioned in Section 1, including linear programs, e.g., MAP LP relaxation [139] and LP decoding [8], and non-smooth optimization, e.g. robust PCA [24, 116]. Another promising scenario for OADM is consensus optimization [19] where distributed local variables are updated separately and reach a global consensus in the long run. More examples can be found in [19] and references therein.

In the sequel, we need the following assumptions:

Assumption 9

(a) For a p-norm $\|\cdot\|_p$, the dual norm of subgradient of $f_t(\mathbf{x})$ is bounded by G_f, i.e., $\|\nabla f_t'(\mathbf{x})\|_q \leq G_f$, where $f_t'(\mathbf{x}) \in \partial f_t(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$ and $\frac{1}{p} + \frac{1}{q} = 1$.

(b) The Bregman divergence B_ϕ is defined on an α-strongly convex function ϕ with respect to a p-norm $\|\cdot\|_p$, i.e., $B_\phi(\mathbf{u},\mathbf{v}) \geq \frac{\alpha}{2}\|\mathbf{x}-\mathbf{y}\|_p^2$ where $\alpha > 0$.

(c) $\mathbf{x}_1 = \mathbf{0}, \mathbf{y}_1 = \mathbf{0}, \mathbf{z}_1 = \mathbf{0}$. For any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, $B_\phi(\mathbf{x}^*, \mathbf{x}_1) \leq D_{\mathbf{x}}^2$, $\|\mathbf{z}^* - \mathbf{z}_1\|_2 \leq D_{\mathbf{z}}$.

(d) $g(\mathbf{z}_1) = 0$ and $g(\mathbf{z}) \geq 0$.

(e) For any t, $f_t(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{z}^*) + g(\mathbf{z}^*)) \geq -F$, where F is a positive constant.

In Assumption 9, (a) and (b) are in general required in the online learning setting [225, 53, 210]. (c) and (d) are simply for the ease of exposition of regret bounds and is commonly assumed for composite objective [53, 210], e.g., g is a regularizer in machine learning. We may assume the convex sets of $\mathbf{x}$ and $\mathbf{z}$ are bounded [225, 75] in (c). To obtain a sublinear regret bound for constraint violation, we need (e), which is true if functions are bounded from below or Lipschitz continuous in the convex set [128].

6.3 Regret Analysis for OADM

We consider two types of regret in OADM. The first type is the regret of the objective based on splitting variables, i.e., R_1 defined in (6.9). Aside from using splitting variables, R_1 is the standard regret in the online learning setting. The second is the regret of the constraint violation R^c defined in (6.10). We establish sublinear regret bounds for several cases whether f_t and g are strongly convex or not.

6.3.1 General Convex Functions

The following establishes the regret bounds for OADM for general convex functions.

Theorem 19 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by OADM (6.6)-(6.8) and let Assumption 9 hold. For any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$ and $\rho = \sqrt{T}$, we have*

$$R_1(T) \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \sqrt{T}}{2} + \frac{\sqrt{2} G_f D_{\mathbf{x}} \sqrt{T}}{\sqrt{\alpha}} \,, \tag{6.12}$$

$$R^c(T) \leq \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2\sqrt{2} D_{\mathbf{x}} G_f}{\sqrt{\alpha}} + 2F\sqrt{T} \,. \tag{6.13}$$

Proof: Since $\mathbf{x}_{t+1}$ minimizes (6.6), we have

$$0 \in \partial f_t(\mathbf{x}_{t+1}) + \mathbf{A}^T\mathbf{y}_t + \rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t - \mathbf{c}) + \eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t)) \,. \tag{6.14}$$

Rearranging the terms and using (6.8) give the subgradient of $f_t(\mathbf{x}_{t+1})$,

$$-\mathbf{A}^T(\mathbf{y}_{t+1} + \rho(\mathbf{B}\mathbf{z}_t - \mathbf{B}\mathbf{z}_{t+1})) - \eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t)) \in \partial f_t(\mathbf{x}_{t+1}) \tag{6.15}$$

Compared to (3.7) in Lemma 6, the additional terms introduced by Bregman divergence are included in the subgradient. Therefore, replacing f by f_t in Lemma 7 and adding the terms $-\eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t))$, we have

$$\begin{aligned}
& f_t(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) \\
& \leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \\
& - \eta\langle\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle \,.
\end{aligned} \tag{6.16}$$

Using the three point property of Bregman divergence, the last term can be written as

$$-\langle\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle = B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t) \,. \tag{6.17}$$

Let $f'_t(\mathbf{x}_t) \in \partial f_t(\mathbf{x}_t)$. According to the Fenchel-Young's inequality [165], i.e., $2|\langle\mathbf{x}, \mathbf{y}\rangle| \leq \|\mathbf{x}\|_q^2 + \|\mathbf{y}\|_p^2$, we have

$$\begin{aligned}
f_t(\mathbf{x}_t) - f_t(\mathbf{x}_{t+1}) \leq \langle f'_t(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}_{t+1}\rangle &= \langle\frac{1}{\sqrt{\alpha\eta}}f'_t(\mathbf{x}_t), \sqrt{\alpha\eta}(\mathbf{x}_t - \mathbf{x}_{t+1})\rangle \\
&\leq \frac{1}{2\alpha\eta}\|f'_t(\mathbf{x}_t)\|_q^2 + \frac{\alpha\eta}{2}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|_p^2 \,.
\end{aligned} \tag{6.18}$$

Recalling $B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t) \geq \frac{\alpha}{2}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|_p^2$ and combining (6.16)-(6.18), we have

$$\begin{aligned}
& f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) \\
& \leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \\
& \quad + \frac{1}{2\alpha\eta}\|f'_t(\mathbf{x}_t)\|_q^2 + \eta(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) \,.
\end{aligned} \tag{6.19}$$

From Assumption 9, $g(\mathbf{z}) \geq 0$ and $g(\mathbf{z}_1) = 0$ for $\mathbf{z}_1 = \mathbf{0}$, $R_1(T)$ is bounded as follows :

$$R_1(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) + g(\mathbf{z}_1) - g(\mathbf{z}_{T+1})$$

$$
\begin{aligned}
&\leq \frac{1}{2\rho}(\|\mathbf{y}_1\|_2^2 - \|\mathbf{y}_{T+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_1\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{T+1}\|_2^2) \\
&\quad + \eta(B_\phi(\mathbf{x}^*, \mathbf{x}_1) - B_\phi(\mathbf{x}^*, \mathbf{x}_{T+1})) + \frac{1}{2\alpha\eta}\sum_{t=1}^{T}\|f_t'(\mathbf{x}_t)\|_q^2 \\
&\leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2} + \eta D_{\mathbf{x}}^2 + \frac{G_f^2 T}{2\alpha\eta}\ . \qquad (6.20)
\end{aligned}
$$

Setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$ and $\rho = \sqrt{T}$ yields (6.12).

Now we prove (6.13). Rearranging the terms in (6.16), we have

$$
\begin{aligned}
\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 &\leq \frac{2F}{\rho} + \frac{1}{\rho^2}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + (\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \\
&\quad + \frac{2\eta}{\rho}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t))\ . \qquad (6.21)
\end{aligned}
$$

Letting Assumption 9 hold and summing over t from 1 to T, we have

$$
\begin{aligned}
&\sum_{t=1}^{T}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \\
&\leq \frac{2FT}{\rho} + \frac{1}{\rho^2}(\|\mathbf{y}_1\|_2^2 - \|\mathbf{y}_{T+1}\|_2^2) + (\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_1\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{T+1}\|_2^2) \\
&\quad + \frac{2\eta}{\rho}(B_\phi(\mathbf{x}^*, \mathbf{x}_1) - B_\phi(\mathbf{x}^*, \mathbf{x}_{T+1})) \\
&\leq \frac{2FT}{\rho} + \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2\eta}{\rho} D_{\mathbf{x}}^2\ . \qquad (6.22)
\end{aligned}
$$

Setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$ and $\rho = \sqrt{T}$, we have (6.13) by using Lemma 9. ■

Note the bounds are achieved without any explicit assumptions on $\mathbf{A}, \mathbf{B}, \mathbf{c}$.[1] The subgradient of f_t is required to be bounded, but the subgradient of g is not necessarily bounded. Thus, the bounds hold for the case where g is an indicator function of a convex set. Compared to regret bound for COMID which is $\frac{G_f D_{\mathbf{x}}\sqrt{T}}{\sqrt{\alpha}}$ [52], the regret bound for the objective of ADMM has an additional term $\frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2\sqrt{T}}{2}$ which is for the splitting variable $\mathbf{z}$. In addition to the $O(\sqrt{T})$ regret bound, OADM achieves the $O(\sqrt{T})$ bound for the constraint violation, which is not considered in the start-of-the-art online learning algorithms [52, 53, 210], since they do not explicitly handle linear constraint of the form $\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}$. In fact, the bound for constraint violation could be reduced to a constant if $\mathbf{y}_t$ is assumed to be bounded (see Assumption 5), which is shown in the following theorem.

[1] We do assume that $\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}$ is feasible.

Theorem 20 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by OADM. Assume that $\|\mathbf{y}_t\|_2 \leq D$. Setting $\rho = \sqrt{T}$, then*

$$\sum_{t=1}^{T} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \leq 4D^2 \; . \tag{6.23}$$

Proof: According to (6.8), we have

$$\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 = \|\frac{1}{\rho}(\mathbf{y}_{t+1} - \mathbf{y}_t)\|_2^2 \leq \frac{2}{\rho^2}(\|\mathbf{y}_{t+1}\|_2^2 + \|\mathbf{y}_t\|_2^2) \leq \frac{4D^2}{\rho^2} \; . \tag{6.24}$$

Summing over t from 1 to T and setting $\rho = \sqrt{T}$ yield (6.23). ■

6.3.2 Strongly Convex Functions

We assume both $f_t(\mathbf{x})$ and g are strongly convex. Specifically, we assume $f_t(\mathbf{x})$ is β_1-strongly convex with respect to a differentiable convex function ϕ, i.e.,

$$f_t(\mathbf{x}^*) \geq f_t(\mathbf{x}) + \langle f_t'(\mathbf{x}), \mathbf{x}^* - \mathbf{x}\rangle + \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}) \; , \tag{6.25}$$

where $f_t'(\mathbf{x})$ denotes the subgradient of f_t at $\mathbf{x}$ and $\beta_1 > 0$. Assume g is a β_2-strongly convex function, i.e.,

$$g(\mathbf{z}^*) \geq g(\mathbf{z}) + \langle g'(\mathbf{z}), \mathbf{z}^* - \mathbf{z}\rangle + \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}\|_2^2 \; , \tag{6.26}$$

where $g'(\mathbf{z})$ denotes the subgradient of g at $\mathbf{z}$ and $\beta_2 > 0$.

Instead of using fixed ρ and η, we allow them to change over time, i.e., ρ_t and η_t, which is fairly standard in the proof of logarithmic regret bounds [75, 53, 210] where the curvature of a sequence of strongly convex functions f_t is considered. The following theorem establishes logarithmic regret bounds for R_1 as well as R^c.

Theorem 21 *Let Assumption 9 hold. Assume $f_t(\mathbf{x})$ and g are strongly convex given in (6.25) and (6.26). Setting $\eta_t = \beta_1 t, \rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$, we have*

$$R_1(T) \leq \frac{G_f^2}{2\alpha\beta_1}\log(T+1) + \frac{\beta_2 D_{\mathbf{z}}^2}{2} + \beta_1 D_{\mathbf{x}}^2 \; , \tag{6.27}$$

$$R^c(T) \leq \frac{2F\lambda_{\max}^{\mathbf{B}}}{\beta_2}\log(T+1) + \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2\beta_1\lambda_{\max}^{\mathbf{B}} D_{\mathbf{x}}^2}{\beta_2} \; . \tag{6.28}$$

Proof: Assume $f_t(\mathbf{x})$ and g are strongly convex (6.25)-(6.26). Let $\mathbf{x}$ be $\mathbf{x}_{t+1}$ and $\mathbf{z}$ be $\mathbf{z}_{t+1}$ in (6.25)-(6.26) respectively. Adding them together and rearranging the terms give

$$\begin{aligned}&f_t(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*))\\ &\leq \langle f_t'(\mathbf{x}_{t+1}), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) + \langle g'(\mathbf{z}_{t+1}), \mathbf{z}_{t+1} - \mathbf{z}^*\rangle - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 .\end{aligned} \tag{6.29}$$

Compared to the general convex case in Theorem 19, the right hand side has two additional strongly convex terms. (6.29) can be obtained by letting ρ, η be ρ_{t+1}, η_{t+1} respectively in (6.16) and adding the two strongly convex term as follows:

$$\begin{aligned}&f_t(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*))\\ &\leq \frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho_{t+1}}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2)\\ &+ \eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t)) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 .\end{aligned} \tag{6.30}$$

Let η be η_{t+1} in (6.18). Adding to (6.30) and ignoring the negative term $-\frac{\rho_{t+1}}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$, we have

$$\begin{aligned}&f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*))\\ &\leq \frac{1}{\eta_{t+1}}\|f_t'(\mathbf{x}_t)\|_*^2 + \frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2)\\ &- \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 + (\eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \eta_{t+1} B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t) .\end{aligned} \tag{6.31}$$

Summing over t from 1 to T, we have

$$\begin{aligned}R_1(T) \leq &\frac{1}{2\alpha}\sum_{t=1}^{T}\frac{1}{\eta_{t+1}}\|f_t'(\mathbf{x}_t)\|_*^2 + \sum_{t=1}^{T}\frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2)\\ &+ \sum_{t=1}^{T}(\frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2)\\ &+ \sum_{t=1}^{T}(\eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) .\end{aligned} \tag{6.32}$$

Assuming ρ_t is non-decreasing, we have

$$\sum_{t=1}^{T} \frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) \leq \frac{1}{2\rho_2}\|\mathbf{y}_1\|_2^2 = 0 \ . \tag{6.33}$$

Using $\|\mathbf{Bz}^* - \mathbf{Bz}_{t+1}\|_2^2 \leq \lambda_{\max}^{\mathbf{B}}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2$ and setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$, we have

$$\begin{aligned}
&\sum_{t=1}^{T} \left[\rho_{t+1}(\|\mathbf{Bz}^* - \mathbf{Bz}_t\|_2^2 - \|\mathbf{Bz}^* - \mathbf{Bz}_{t+1}\|_2^2) - \beta_2\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2\right] \\
&\leq \sum_{t=1}^{T} \left[\rho_{t+1}(\|\mathbf{Bz}^* - \mathbf{Bz}_t\|_2^2 - \|\mathbf{Bz}^* - \mathbf{Bz}_{t+1}\|_2^2) - \frac{\beta_2}{\lambda_{\max}^{\mathbf{B}}}\|\mathbf{Bz}^* - \mathbf{Bz}_{t+1}\|_2^2\right] \\
&\leq \rho_2\|\mathbf{Bz}^* - \mathbf{Bz}_1\|_2^2 + \sum_{t=2}^{T}\|\mathbf{Bz}^* - \mathbf{Bz}_t\|_2^2(\rho_{t+1} - \rho_t - \frac{\beta_2}{\lambda_{\max}^{\mathbf{B}}}) \\
&= 2\beta_2 D_{\mathbf{z}}^2 \ ,
\end{aligned} \tag{6.34}$$

where the last equality uses the Assumption 9. Similarly, setting $\eta_t = \beta_1 t$, the last term in (6.32) can be rewritten as

$$\begin{aligned}
&\sum_{t=1}^{T} [\eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})] \\
&= \eta_2 B_\phi(\mathbf{x}^*, \mathbf{x}_1) + \sum_{t=2}^{T} B_\phi(\mathbf{x}^*, \mathbf{x}_t)(\eta_{t+1} - \eta_t - \beta_1) - \eta_{T+1}B_\phi(\mathbf{x}^*, \mathbf{x}_{T+1}) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_{T+1}) \\
&\leq \eta_2 B_\phi(\mathbf{x}^*, \mathbf{x}_1) + \sum_{t=2}^{T} B_\phi(\mathbf{x}^*, \mathbf{x}_t)(\eta_{t+1} - \eta_t - \beta_1) \\
&= 2\beta_1 D_{\mathbf{x}}^2 \ .
\end{aligned} \tag{6.35}$$

Setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}, \eta_t = \beta_1 t$ and combining (6.32), (6.33), (6.34) and (6.35), we have

$$R_1(T) \leq \frac{G_f^2}{2\alpha}\sum_{t=1}^{T}\frac{1}{\beta_1(t+1)} + \beta_2 D_{\mathbf{z}}^2 + 2\beta_1 D_{\mathbf{x}}^2 \ . \tag{6.36}$$

Applying $\sum_{t=1}^{T}\frac{1}{t+1} \leq \int_{t=0}^{T}\frac{1}{t+1}dt = \log(T+1)$ gives (6.27).

Now we prove (6.28). Rearranging terms in (6.30), we have

$$\|\mathbf{Ax}_{t+1} + \mathbf{Bz}_t - \mathbf{c}\|_2^2 \leq \frac{2F}{\rho_{t+1}} + \frac{1}{\rho_{t+1}^2}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + (\|\mathbf{Bz}^* - \mathbf{Bz}_t\|_2^2 - \|\mathbf{Bz}^* - \mathbf{Bz}_{t+1}\|_2^2)$$

$$+ \frac{2\eta_{t+1}}{\rho_{t+1}}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1}) - B_\phi(\mathbf{x}_{t+1}, \mathbf{x}_t)) \ . \tag{6.37}$$

Letting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$ and $\eta_t = \beta_1 t$ and summing over t from 0 to T, we have

$$\begin{aligned}
&\sum_{t=1}^{T} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \\
&\le 2F\sum_{t=1}^{T}\frac{1}{\rho_{t+1}} + \sum_{t=1}^{T}\frac{1}{\rho_{t+1}^2}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + (\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_0\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{T+1}\|_2^2) \\
&+ \sum_{t=1}^{T}\frac{2\eta_{t+1}}{\rho_{t+1}}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) \\
&\le \frac{2F\lambda_{\max}^{\mathbf{B}}\log(T+1)}{\beta_2} + \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2\beta_1\lambda_{\max}^{\mathbf{B}} D_{\mathbf{x}}^2}{\beta_2} \ .
\end{aligned} \tag{6.38}$$

We use (6.33) in the last inequality. According to Lemma 9, we have (6.28). ■

To guarantee logarithmic regret bounds for both objective and constraints violation, OADM requires both f_t and g to be strongly convex. FOBOS, COMID, and RDA only require g to be strongly convex although they do not consider linear constraints explicitly. Further, the logarithmic regret bounds for the constraints violation could reduce to constant bound if assuming $\mathbf{y}_t$ is bounded.

Theorem 22 *Let the sequences $\{\mathbf{x}_t, \mathbf{z}_t, \mathbf{y}_t\}$ be generated by OADM and $\|\mathbf{y}_t\|_2 \le D$. Setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$, then*

$$\sum_{t=1}^{T} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \le \frac{2\pi D^2 {\lambda_{\max}^{\mathbf{B}}}^2}{3\beta_2^2} \ . \tag{6.39}$$

Proof: Replacing ρ by ρ_{t+1} in (6.24) and summing over t from 1 to T, we have

$$\sum_{t=1}^{T} \|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \le \sum_{t=1}^{T}\frac{4D^2}{\rho_{t+1}^2} \ . \tag{6.40}$$

Setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$ and using $\sum_{t=1}^{T}\frac{1}{t^2} \le \frac{\pi}{6}$ complete the proof. ■

6.4 Regret Analysis for OADM with $\eta = 0$

We analyze the regret bound when $\eta = 0$. In this case, OADM has the same updates as ADM except f_t is changing over time. The $\mathbf{x}$-update only including the quadratic penalty term is easier to solve than the one with an additional Bregman divergence, particularly when the Bregman divergence is not a quadratic function. Without a Bregman divergence to keep two consecutive iterates of $\mathbf{x}$ close, the quadratic penalty term is qualified for this task through variable $\mathbf{z}$. We consider $\mathbf{z}_t$ to be the key primal variable, and compute $\hat{\mathbf{x}}_t$ using $\mathbf{z}_t$ so that $\mathbf{A}\hat{\mathbf{x}}_t + \mathbf{B}\mathbf{z}_t = \mathbf{c}$. Therefore, we use the regret bound R_2 defined in (6.11). While $\{\hat{\mathbf{x}}_t, \mathbf{z}_t\}$ satisfies the equality constraint, $\{\mathbf{x}_t, \mathbf{z}_t\}$ need not satisfy $\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c} = \mathbf{0}$. Therefore, we also consider bounds for R^c as defined in (6.10). A common case we often encounter is when $\mathbf{A} = \mathbf{I}, \mathbf{B} = -\mathbf{I}, \mathbf{c} = \mathbf{0}$, thus $\hat{\mathbf{x}}_t = \mathbf{z}_t$. Consensus optimization is a typical example of this form [19, 15, 141]. In machine learning, many examples like (group) lasso [19, 218] can be reformulated in this way.

In this section, we need additional assumptions. In Assumption 9 (a), we specify the dual norm $\|\cdot\|_q$ to be ℓ_2, i.e., $\|f_t(\mathbf{x})\|_2 \leq G_f$. To guarantee that $\mathbf{A}\hat{\mathbf{x}}_t + \mathbf{B}\mathbf{z}_t = \mathbf{c}, \mathbf{A} \in \mathbb{R}^{m \times n_1}$ is feasible, the equality constraint, in particular, implicitly requires the assumption $m \leq n_1$. On the other hand, to establish a bound for R_2, $\mathbf{A}$ should be full-column rank, i.e., $rank(\mathbf{A}) = n_1$. Therefore, we need the following assumption in this scenario:

Assumption 10 $\mathbf{A}$ *is a square and full rank matrix, i.e.,* $\mathbf{A}$ *is invertible. Let* $\lambda_{\min}^{\mathbf{A}}$ *be the smallest eigenvalue of* $\mathbf{A}\mathbf{A}^T$*, then* $\lambda_{\min}^{\mathbf{A}} > 0$*.*

Assumption 10 is satisfied in most examples like lasso and consensus optimization. Considering the subgradient of f_t given in (3.6), if there always exists a vector $\mathbf{v}_t$ such that $-\mathbf{A}^T\mathbf{v}_t \in \partial f_t(\mathbf{x}_t)$, Assumption 10 can be safely removed under the implicit assumption that $\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}$ is feasible.

6.4.1 General Convex Functions

The following theorem shows the regret bounds for R_2 as well as R^c.

Theorem 23 *Let* $\eta = 0$ *in OADM. Let Assumption 9 and 10 hold. For any* $\mathbf{x}^*, \mathbf{z}^*$ *satisfying*

$\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, *setting* $\rho = \frac{G_f\sqrt{T}}{D_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}}$, *we have*

$$R_2(T) \leq \frac{G_f D_{\mathbf{z}}\sqrt{\lambda_{\max}^{\mathbf{B}}}}{\sqrt{\lambda_{\min}^{\mathbf{A}}}}\sqrt{T} \ , \tag{6.41}$$

$$R^c(T) \leq \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2FD_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}} T}}{G_f} \ . \tag{6.42}$$

Proof: Replacing f by f_t in Lemma 7, we have

$$\begin{aligned}&f_t(\mathbf{x}_{t+1}) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))\\ &\leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \ .\end{aligned} \tag{6.43}$$

Let $f_t'(\hat{\mathbf{x}}_t) \in \partial f_t(\hat{\mathbf{x}}_t)$. Recalling $\mathbf{A}\hat{\mathbf{x}}_t + \mathbf{B}\mathbf{z}_t = \mathbf{c}$, then

$$\begin{aligned}&f_t(\hat{\mathbf{x}}_t) - f_t(\mathbf{x}_{t+1}) \leq \langle f_t'(\hat{\mathbf{x}}_t), \hat{\mathbf{x}}_t - \mathbf{x}_{t+1}\rangle = \langle (\mathbf{A}^{-1})^T f_t'(\hat{\mathbf{x}}_t), \mathbf{A}\hat{\mathbf{x}}_t - \mathbf{A}\mathbf{x}_{t+1}\rangle\\ &= -\langle (\mathbf{A}^{-1})^T f_t'(\hat{\mathbf{x}}_t), \mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\rangle \leq \frac{1}{2\lambda_{\min}^{\mathbf{A}}\rho}\|f_t'(\hat{\mathbf{x}}_t)\|_2^2 + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \ .\end{aligned} \tag{6.44}$$

Adding to (6.43) gives

$$\begin{aligned}&f_t(\hat{\mathbf{x}}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*))\\ &\leq \frac{1}{2\lambda_{\min}^{\mathbf{A}}\rho}\|f_t'(\hat{\mathbf{x}}_t)\|_2^2 + \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \ .\end{aligned} \tag{6.45}$$

Letting the assumptions hold, $R_2(T)$ is bounded as:

$$\begin{aligned}R_2(T) &\leq \sum_{t=1}^{T}[f_t(\hat{\mathbf{x}}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*))]\\ &\leq \frac{1}{2\lambda_{\min}^{\mathbf{A}}\rho}\sum_{t=1}^{T}\|f_t'(\hat{\mathbf{x}}_t)\|_2^2 + \frac{1}{2\rho}(\|\mathbf{y}_1\|_2^2 - \|\mathbf{y}_{T+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_1\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{T+1}\|_2^2)\\ &\leq \frac{G_f^2 T}{2\lambda_{\min}^{\mathbf{A}}\rho} + \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2\rho}{2} \ .\end{aligned} \tag{6.46}$$

Setting $\rho = \frac{G_f\sqrt{T}}{D_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}}$ yields (6.41).

Now we prove (6.42). Rearranging the terms in (6.43), we have

$$\|\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}\|_2^2 \leq \frac{2F}{\rho}+\frac{1}{\rho^2}(\|\mathbf{y}_t\|_2^2-\|\mathbf{y}_{t+1}\|_2^2)+(\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_t\|_2^2-\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_{t+1}\|_2^2)\ . \tag{6.47}$$

Letting the assumptions hold and summing over t from 1 to T, we have

$$\begin{aligned}&\sum_{t=1}^{T}\|\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}\|_2^2\\&\leq \frac{2FT}{\rho}+\frac{1}{\rho^2}(\|\mathbf{y}_1\|_2^2-\|\mathbf{y}_{T+1}\|_2^2)+(\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_1\|_2^2-\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_{T+1}\|_2^2)\\&\leq \frac{2FT}{\rho}+\lambda_{\max}^{\mathbf{B}}D_{\mathbf{z}}^2\ .\end{aligned} \tag{6.48}$$

Setting $\rho=\frac{G_f\sqrt{T}}{D_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}}$ and using Lemma 9 give (6.42). ■

The following theorem shows that R^c has a constant bound when assuming $\|\mathbf{y}\|_2\leq D^2$.

Theorem 24 *Let the sequences $\{\mathbf{x}_t,\mathbf{z}_t,\mathbf{y}_t\}$ be generated by OADM with $\eta=0$. Let Assumption 10 hold. Assuming $\|\mathbf{y}_t\|_2\leq D^2$ and setting $\rho=\frac{G_f\sqrt{T}}{D_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}}$, we have*

$$R^c(T)\leq \frac{2D_{\mathbf{z}}^2\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}{G_f^2}(D^2+\frac{G_f^2}{\lambda_{\min}^{\mathbf{A}}})\ . \tag{6.49}$$

Proof: Let f be f_t in (3.6). Define

$$f_t'(\mathbf{x}_{t+1})=-(\mathbf{A}^T\mathbf{y}_t+\rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}))\ . \tag{6.50}$$

Multiplying both sides by $(\mathbf{A}^T)^{-1}$ gives

$$(\mathbf{A}^T)^{-1}f_t'(\mathbf{x}_{t+1})=-(\mathbf{y}_t+\rho(\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}))\ . \tag{6.51}$$

Rearranging the terms, we have

$$\begin{aligned}\|\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}\|_2^2&=\frac{1}{\rho^2}\|\mathbf{y}_t+(\mathbf{A}^T)^{-1}f_t'(\mathbf{x}_{t+1})\|_2^2\\&\leq\frac{2}{\rho^2}(\|\mathbf{y}_t\|_2^2+\|(\mathbf{A}^T)^{-1}f_t'(\mathbf{x}_{t+1})\|_2^2)\\&\leq\frac{2}{\rho^2}(D^2+\frac{G_f^2}{\lambda_{\min}^{\mathbf{A}}})\ .\end{aligned} \tag{6.52}$$

Summing over t from 1 to T and setting $\rho = \frac{G_f\sqrt{T}}{D_{\mathbf{z}}\sqrt{\lambda_{\min}^{\mathbf{A}}\lambda_{\max}^{\mathbf{B}}}}$, we have (6.49) according to Lemma 2. ■

Without requiring an additional Bregman divergence, R_2 achieves the same $\sqrt{T}$ bound as R_1. While R_1 depends on $\mathbf{x}_t$ which may not stay in the feasible set, R_2 is defined on $\hat{\mathbf{x}}_t$ which always satisfies the equality constraint. The corresponding algorithm requires finding $\hat{\mathbf{x}}_t$ in each iteration such that $\mathbf{A}\hat{\mathbf{x}}_t = \mathbf{c} - \mathbf{B}\mathbf{z}_t$, which involves solving a linear system. The algorithm will be efficient in some settings, e.g., consensus optimization where $\mathbf{A} = \mathbf{I}$.

6.4.2 Strongly Convex Functions

If $g(\mathbf{z})$ is a β_2-strongly convex function given in (6.26), we show that R_2 and R^c have logarithmic bounds.

Theorem 25 *Let $\eta = 0$ in OADM. Assume that $g(\mathbf{z})$ is β_2-strongly convex and Assumption 9 and 10 hold. Setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$, we have*

$$R_2(T) \le \frac{G_f^2\lambda_{\max}^{\mathbf{B}}}{2\lambda_{\min}^{\mathbf{A}}\beta_2}(\log(T+1)) + \beta_2 D_{\mathbf{z}}^2 \,, \tag{6.53}$$

$$R^c(T) \le \lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 + \frac{2F\lambda_{\max}^{\mathbf{B}}}{\beta_2}\log(T+1) \,. \tag{6.54}$$

Proof: Assuming $g(\mathbf{z})$ is strongly convex (6.26), we can show the regret bound by replacing ρ by ρ_{t+1} and subtracting the strongly convex term $\frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2$ in (6.45), i.e.,

$$\begin{aligned} f_t(\hat{\mathbf{x}}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) &\le \frac{1}{2\lambda_{\min}^{\mathbf{A}}\rho_{t+1}}\|f_t'(\hat{\mathbf{x}}_t)\|_2^2 + \frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) \\ + \frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) &- \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 \,. \end{aligned} \tag{6.55}$$

Summing over t from 1 to T, we have

$$\begin{aligned} R_2(T) \le &\frac{G_f^2}{2\lambda_{\min}^{\mathbf{A}}}\sum_{t=1}^{T}\frac{1}{\rho_{t+1}} + \sum_{t=1}^{T}\frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) \\ &+ \sum_{t=1}^{T}\left[\frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2\right] \,. \end{aligned} \tag{6.56}$$

Using (6.33), (6.34) and setting $\rho_t = \beta_2 t/\lambda_{\max}^{\mathbf{B}}$, we get (6.53) by applying $\sum_{t=1}^{T}\frac{1}{t+1} \le \log(T+1)$.

Now we prove (6.54). Replacing ρ by ρ_{t+1} in (6.47), we have

$$\|\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}\|_2^2 \leq \frac{2F}{\rho_{t+1}}+\frac{1}{\rho_{t+1}^2}(\|\mathbf{y}_t\|_2^2-\|\mathbf{y}_{t+1}\|_2^2)+(\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_t\|_2^2-\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_{t+1}\|_2^2)\,. \tag{6.57}$$

Letting the assumptions hold and summing over t from 0 to T, we have

$$\begin{aligned}&\sum_{t=1}^{T}\|\mathbf{A}\mathbf{x}_{t+1}+\mathbf{B}\mathbf{z}_t-\mathbf{c}\|_2^2\\ &\leq 2F\sum_{t=1}^{T}\frac{1}{\rho_{t+1}}+\sum_{t=1}^{T}\frac{1}{\rho_{t+1}^2}(\|\mathbf{y}_t\|_2^2-\|\mathbf{y}_{t+1}\|_2^2)+(\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_1\|_2^2-\|\mathbf{B}\mathbf{z}^*-\mathbf{B}\mathbf{z}_{T+1}\|_2^2)\\ &\leq 2F\sum_{t=1}^{T}\frac{1}{\rho_{t+1}}+\lambda_{\max}^{\mathbf{B}}D_{\mathbf{z}}^2\,.\end{aligned} \tag{6.58}$$

We use (6.33) in the last inequality. Setting $\rho_t=\beta_2 t/\lambda_{\max}^{\mathbf{B}}$ and using Lemma 9 give (6.54). ■

Similar as the case of general convex functions, the logarithmic regret bound for constraint violation can also be reduced to a constant bound, as shown in the following theorem.

Theorem 26 *Let $\eta=0$ in OADM. Assume that $g(\mathbf{z})$ is β_2-strongly convex and Assumption 10 hold. Assuming $\|\mathbf{y}_t\|_2\leq D$ and setting $\rho_t=\beta_2 t/\lambda_{\max}^{\mathbf{B}}$, we have*

$$R^c(T)\leq \frac{\pi{\lambda_{\max}^{\mathbf{B}}}^2}{3\beta_2^2}\left(D^2+\frac{G_f^2}{\lambda_{\min}^{\mathbf{A}}}\right) \tag{6.59}$$

Proof: Setting $\rho_t=\beta_2 t/\lambda_{\max}^{\mathbf{B}}$ in (6.52), summing over t from 1 to T and using $\sum_{t=1}^{T}\frac{1}{t^2}\leq\frac{\pi}{6}$ complete the proof. ■

Theorem 26 shows that OADM can achieve the logarithmic regret bound without requiring f_t to be strongly convex, which is in line with other online learning algorithms for composite objectives.

6.5 Further Discussions

In this section, we discuss several variants of the $\mathbf{x}$ update in OADM which can lead to efficient updates and show the stochastic convergence rates. The connection to the related work is presented. We also show that OADM can serve as projection-free online learning.

6.5.1 Inexact ADMM Updates ($\eta > 0$)

In OADM ($\eta > 0$), since the $\mathbf{x}$ update (6.6) involves the function f_t, the quadratic penalty term and a Bregman divergence, it may be computationally expensive to solve it exactly. We consider several variants which solve the $\mathbf{x}$ update inexactly through the linearization of some terms. The inexact updates can be efficient, and include mirror descent algorithm (MDA) and composite objective mirror descent (COMID) as special cases.

Case 1: Linearization of the quadratic penalty term The linearization of the quadratic penalty term in (6.6) can be done by removing $\|\mathbf{A}\mathbf{x}\|_2^2$ as follows:

$$\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 - \|\mathbf{A}(\mathbf{x} - \mathbf{x}_t)\|_2^2 = 2\langle \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c}, \mathbf{A}\mathbf{x}\rangle + \|\mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 - \|\mathbf{A}\mathbf{x}_t\|_2^2 \ .$$

Let $B_\phi(\mathbf{x}, \mathbf{x}_t) = B_\varphi(\mathbf{x}, \mathbf{x}_t) - \frac{\rho}{2\eta}\|\mathbf{A}(\mathbf{x} - \mathbf{x}_t)\|_2^2$ in (6.6), where B_φ is a Bregman divergence and the quadratic term is used to linearize the quadratic penalty term. Removing constant terms, (6.6) becomes

$$\mathbf{x}_{t+1} = \operatorname{argmin}_{\mathbf{x}} f_t(\mathbf{x}) + \langle \mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c}), \mathbf{A}\mathbf{x}\rangle + \eta B_\varphi(\mathbf{x}, \mathbf{x}_t) \ . \tag{6.60}$$

This case mainly solves the problem caused by $\mathbf{A}$, e.g., $\mathbf{A}\mathbf{x}$ makes $\mathbf{x}$ nonseparable. Several problems have been benefited from the linearization of quadratic term [48], e.g., f is ℓ_1 loss function [74] and projection onto the unit simplex or ℓ_1 ball [51].

Since $B_\phi(\mathbf{x}, \mathbf{x}_t) \geq \frac{\alpha}{2}\|\mathbf{x} - \mathbf{x}_t\|_2^2$ is required for the analysis in Section 6.3, B_φ should be chosen to satisfy that condition. Note

$$B_\phi(\mathbf{x}, \mathbf{x}_t) = B_\varphi(\mathbf{x}, \mathbf{x}_t) - \frac{\rho}{2\eta}\|\mathbf{A}(\mathbf{x} - \mathbf{x}_t)\|_2^2 \geq B_\varphi(\mathbf{x}, \mathbf{x}_t) - \frac{\rho\lambda_{\max}^{\mathbf{A}}}{2\eta}\|\mathbf{x} - \mathbf{x}_t\|_2^2 \ . \tag{6.61}$$

Therefore, as long as $B_\varphi(\mathbf{x}, \mathbf{x}_t) \geq \frac{\rho\lambda_{\max}^{\mathbf{A}}/\eta + \alpha}{2}\|\mathbf{x} - \mathbf{x}_t\|_2^2$, the assumption 9(b) holds, meaning Theorem 19 and 21 hold for Case 1.

Case 2: Linearization of function f_t This case is particularly useful when the difficulty of solving (6.6) is caused by $f_t(\mathbf{x})$, e.g., when f_t is a logistic loss function. Linearizing the function f_t at $\mathbf{x}_t$ in (6.6), we have

$$\mathbf{x}_{t+1} = \operatorname{argmin}_{\mathbf{x}} \langle f_t'(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t\rangle + \frac{\rho}{2}\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \eta B_\phi(\mathbf{x}, \mathbf{x}_t) \ . \tag{6.62}$$

The updated is called inexact ADMM update if ϕ is a quadratic function [19]. In the Appendix 6.A, we show Theorem 19 and 21 continue to hold in this case.

Case 3: Mirror Descent In this case, we linearize both the function and the quadratic term, which can be done by choosing $B_{\phi}(\mathbf{x},\mathbf{x}_t) = B_{\varphi}(\mathbf{x},\mathbf{x}_t) - \frac{\rho}{2\eta}\|\mathbf{A}(\mathbf{x}-\mathbf{x}_t)\|_2^2$ in Case 2. Combining the results in Case 1 and 2, (6.6) becomes the following MDA-type update:

$$\mathbf{x}_{t+1} = \mathrm{argmin}_{\mathbf{x}} \langle F_t(\mathbf{x}_t), \mathbf{x}\rangle + \eta B_{\varphi}(\mathbf{x},\mathbf{x}_t) \,, \tag{6.63}$$

where $F_t(\mathbf{x}_t) = f_t'(\mathbf{x}_t) + \mathbf{A}^T\{\mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c})\}$, which is the gradient of the objective in (6.6). Assuming $B_{\varphi}(\mathbf{x},\mathbf{x}_t) \geq \frac{\rho\lambda_{\max}^{\mathbf{A}}/\eta+\alpha}{2}\|\mathbf{x}-\mathbf{x}_t\|_2^2$ in Case 2, the regret bounds in Theorem 19 and 21 still holds in Case 3.

Case 4: COMID Assume f_t is a composite objective consisting of smooth and nonsmooth part, i.e., $f_t(\mathbf{x}) = f_t^S(\mathbf{x}) + f_t^N(\mathbf{x})$, where f_t^S is the smooth part and f_t^N is the nonsmooth part. Let $B_{\phi}(\mathbf{x},\mathbf{x}_t) = B_{\varphi}(\mathbf{x},\mathbf{x}_t) - \frac{\rho}{2\eta}\|\mathbf{A}(\mathbf{x}-\mathbf{x}_t)\|_2^2$, which is used to linearize the quadratic penalty term. Linearizing the smooth function f_t^S, (6.6) becomes the following COMID-type update:

$$\mathbf{x}_{t+1} = \mathrm{argmin}_{\mathbf{x}}\, f_t^N(\mathbf{x}) + \langle F_t^S(\mathbf{x}_t), \mathbf{x}\rangle + \eta B_{\varphi}(\mathbf{x},\mathbf{x}_t) \,, \tag{6.64}$$

where $F_t^S(\mathbf{x}_t) = \nabla f_t^S(\mathbf{x}_t) + \mathbf{A}^T\{\mathbf{y}_t + \rho(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{z}_t - \mathbf{c})\}$. Applying the analysis in Case 2 on the smooth part, we can get the regret bounds in Theorem 19 and 21.

6.5.2 Stochastic Convergence Rates

In this section, we present the convergence rates for ADMM in the Case 2-4 in Section 6.1 in the stochastic setting, which solves the following stochastic learning problem:

$$\min_{\mathbf{x}\in\mathcal{X},\mathbf{z}\in\mathcal{Z}} \mathbf{E}_{\xi}[f(\mathbf{x},\xi)] + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c} \tag{6.65}$$

$f'(\mathbf{x}_t,\xi_t)$ is an unbiased estimate of $f'(\mathbf{x}_t)$ and $f(\mathbf{x}) = \mathbf{E}f(\mathbf{x},\xi)$. Correspondingly, the $\mathbf{x}$-update in (6.62)-(6.63) uses $f'(\mathbf{x}_t,\xi_t)$ to substitute $f_t'(\mathbf{x}_t)$ and $\nabla f^N(\mathbf{x}_t,\xi_t)$ to substitute $\nabla f_t^N(\mathbf{x}_t)$ in (6.64). The regret bounds for Case 2-4 in Section 6.1 can be converted to convergence rates in the stochastic setting based on known online-stochastic conversion [26, 53, 210]. More specifically, the stochastic convergence rates in expectation can be obtained by simply dividing regret bounds by T. Using martingale concentration results [26, 53, 210], the high probability bounds can also be obtained by applying the Azuma-Hoeffding inequality [3].

Corollary 4 *Let the sequences $\{\mathbf{x}_t,\mathbf{z}_t,\mathbf{y}_t\}$ be generated by stochastic ADM and Assumption 9 hold. Let $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{x}_t$ and $\bar{\mathbf{z}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{z}_t$. For any $\mathbf{x}^*, \mathbf{z}^*$ satisfying $\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{z}^* = \mathbf{c}$, setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$ and $\rho = \sqrt{T}$, we have* *(a) Stochastic convergence rates in expectation*

$$\mathbf{E}\left[f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T)\right] - (f(\mathbf{x}^*) + g(\mathbf{z}^*))] \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{2\sqrt{T}} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{\alpha}\sqrt{T}} , \tag{6.66}$$

$$\mathbf{E}\left[\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2\right] \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{T} + \frac{2\sqrt{2} D_{\mathbf{x}} G_f}{\sqrt{\alpha} T} + \frac{2F}{\sqrt{T}} . \tag{6.67}$$

(b) High probability bounds for stochastic convergence rates

$$P\big(f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \geq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{2\sqrt{T}} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{\alpha}\sqrt{T}} + \varepsilon\big) \leq \exp\left(-\frac{T\alpha\varepsilon^2}{16 D_{\mathbf{x}}^2 G_f^2}\right) , \tag{6.68}$$

$$P\big(\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T - \mathbf{c}\|_2^2 \geq \frac{2F}{\sqrt{T}} + \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{T} + \frac{2\sqrt{2} D_{\mathbf{x}} G_f}{\sqrt{\alpha} T} + \varepsilon\big) \leq \exp\left(-\frac{T\alpha\varepsilon^2}{16 D_{\mathbf{x}}^2 G_f^2}\right) . \tag{6.69}$$

The proof is presented in Appendix 6.B. Compared to the stochastic convergence rates for COMID [52], the stochastic convergence rates for the objective of ADM has an extra term $\frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{2\sqrt{T}}$ which bounds the splitting variable $\mathbf{z}$. For strongly convex functions, we have $O(\frac{\log T}{T})$ stochastic convergence rates by applying the online-stochastic conversion [26, 53, 210] on Theorem 21.

Remark 4 We note that [152] has recently established the stochastic convergence rates for stochastic ADM based on our VI analysis (see Section 2.3), which has the following form in our notation:

$$\mathbf{E}\left[f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + D\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2\right] \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2T} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{T}} + \frac{D^2}{2\rho T} , \tag{6.70}$$

where $\|\mathbf{y}_t\|_2 \leq D$ (see Assumption 5). The bound in (6.70) depends on D^2, which usually is large (see Eq. (3.55)) and thus worse than our results which do not rely on D^2. As a matter of fact, we can show the term D^2 can be safely removed (setting $\alpha = 1$ in (6.106) in Appendix 6.B), i.e.,

$$\mathbf{E}\left[f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2}\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2\right] \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2T} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{T}} . \tag{6.71}$$

However, since $\mathbf{x}_t, \mathbf{z}_t$ are not feasible, $f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*))$ may be negative. As a result, (6.70) or (6.71) may not imply an $O(1/T)$ convergence rate for the equality constraint, in constrast to (6.67) in Corollary 4. Furthermore, if assuming $\|\mathbf{y}_t\|_2 \leq D$, the residual of equality constraint has an $O(1/T)$ convergence rate by dividing by T on both sides of (6.23) in Theorem 20 and using the Jensen's inequality.

6.5.3 Connections to Related Work ($\eta = 0$)

Assume $\eta = 0, \mathbf{A} = \mathbf{I}, \mathbf{B} = -\mathbf{I}, \mathbf{c} = \mathbf{0}$, thus $\mathbf{x} = \mathbf{z}$. Hence, the online optimization problem has the form which is the same as the ones considered in the development of FOBOS [53] and RDA [210]. The three steps of OADM ($\eta = 0$) reduce to

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\{f_t(\mathbf{x}) + \langle \mathbf{y}_t, \mathbf{x} - \mathbf{z}_t \rangle + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}_t\|_2^2\} , \tag{6.72}$$

$$\mathbf{z}_{t+1} = \underset{\mathbf{z}}{\operatorname{argmin}}\{g(\mathbf{z}) + \langle \mathbf{y}_t, \mathbf{x}_{t+1} - \mathbf{z} \rangle + \frac{\rho}{2}\|\mathbf{x}_{t+1} - \mathbf{z}\|_2^2\} , \tag{6.73}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \rho(\mathbf{x}_{t+1} - \mathbf{z}_{t+1}) . \tag{6.74}$$

Let $f_t'(\mathbf{x}_{t+1}) \in \partial f_t(\mathbf{x}), g'(\mathbf{z}_{t+1}) \in \partial g(\mathbf{z})$. The first order optimality conditions for (6.72) and (6.73) give

$$f_t'(\mathbf{x}_{t+1}) + \mathbf{y}_t + \rho(\mathbf{x}_{t+1} - \mathbf{z}_t) = 0 ,$$
$$g'(\mathbf{z}_{t+1}) - \mathbf{y}_t - \rho(\mathbf{x}_{t+1} - \mathbf{z}_{t+1}) = 0 .$$

Adding them together yields

$$\mathbf{z}_{t+1} = \mathbf{z}_t - \frac{1}{\rho}(f_t'(\mathbf{x}_{t+1}) + g'(\mathbf{z}_{t+1})) . \tag{6.75}$$

OADM can be considered as taking the implicit subgradient of f_t and g at the yet to be determined $\mathbf{x}_{t+1}$ and $\mathbf{z}_{t+1}$. FOBOS has the following update [53]:

$$\mathbf{z}_{t+1} = \mathbf{z}_t - \frac{1}{\rho}(f_t'(\mathbf{z}_t) + g'(\mathbf{z}_{t+1})) .$$

FOBOS takes the explicit subgradient of f_t at current $\mathbf{z}_t$. In fact, FOBOS can be considered as a variant of OADM, which linearizes the objective of (6.72) at $\mathbf{z}_t$:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x}}{\operatorname{argmin}}\langle f_t'(\mathbf{z}_t) + \mathbf{y}_t, \mathbf{x} - \mathbf{z}_t \rangle + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}_t\|_2^2 .$$

It has a closed-form solution, i.e., $\mathbf{x}_{t+1} = \mathbf{z}_t - \frac{1}{\rho}(f_t'(\mathbf{z}_t) + \mathbf{y}_t)$. Denote $\mathbf{z}_{t+\frac{1}{2}} = \mathbf{x}_{t+1} + \frac{1}{\rho}\mathbf{y}_t$, then

$$\mathbf{z}_{t+\frac{1}{2}} = \mathbf{z}_t - \frac{1}{\rho} f_t'(\mathbf{z}_t) \ . \tag{6.76}$$

(6.73) is equivalent to the following form:

$$\mathbf{z}_{t+1} = \mathrm{argmin}_{\mathbf{z}}\, g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{z} - \mathbf{z}_{t+\frac{1}{2}}\|_2^2 \ . \tag{6.77}$$

(6.76) and (6.77) form the updates of FOBOS [53]. Furthermore, if $g(\mathbf{z})$ is an indicator function of a convex set Ω, substituting (6.76) into (6.77), we have

$$\mathbf{z}_{t+1} = \mathrm{argmin}_{\mathbf{z}\in\Omega} \frac{\rho}{2}\|\mathbf{z}_t - \frac{1}{\tau} f_t'(\mathbf{z}_t) - \mathbf{z}\|_2^2 = \mathcal{P}_{\mathbf{z}\in\Omega}\left[\mathbf{z}_t - \frac{1}{\tau} f_t'(\mathbf{z}_t)\right] \ ,$$

and we recover projected gradient descent [75].

6.5.4 Projection-free Online Learning

For an online constrained optimization problem, the state-of-the-art methods like OGD, FOBOS and RDA require a full projection onto the constraint set at each round. In many cases, e.g., an intersection of simple constraints, the full projection can be done by alternating projecting onto simple constraints cyclically [25]. In OADM, we can decompose functions and constraints into simpler subproblems by introducing appropriate splitting variables. If the subproblem for each splitting variable is simple enough to yield efficient projection, the full projection onto the whole constraint set can be done by projections onto simple constraints at each round along with the long term equality constraints. Therefore, OADM and its variants can avoid the full projection at each round. Consider the full projection onto $\mathcal{X} \times \mathcal{Z}$, which in general requires alternating projection onto $\mathcal{X}$ and $\mathcal{Z}$ at each round in OGD, FOBOS and RDA. In OADM, by introducing equality constraint $\mathbf{x} = \mathbf{z}$, the constraint set is split into two parts and $\mathbf{x} \in \mathcal{X}$ and $\mathbf{z} \in \mathcal{Z}$. At each round, the primal updates in OADM and its variants project $\mathbf{x}, \mathbf{z}$ onto $\mathcal{X}, \mathcal{Z}$ separately. In the long run, the equality constraint will be satisfied in expectation, thus $\mathbf{x}$ is a feasible solution. Hence, OADM can be considered as a projection-free online learning algorithm.

In [77], the Frank-Wolfe algorithm is used as a projection-free online learning algorithm, which solves a linear optimization at each round and has $O(T^{3/4})$ regret bound. It assumes linear optimization can be done efficiently in the constraint set. Realizing that solving a linear optimization still requires an inner loop algorithm, the authors pose an open problem whether the optimal regret bound can be achieved by performing one iteration of linear-optimization.

We now show how OADM does projection-free online learning with linear constraints, which includes linear programming and quadratic programming as special cases. Formally, we consider the problem

$$\min_{\mathbf{x}} \sum_{t=1}^{T} f_t(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{a}, \mathbf{B}\mathbf{x} \leq \mathbf{b} \ . \tag{6.78}$$

In the setting of OADM, we first introduce an auxiliary variable $\mathbf{z} = \mathbf{B}\mathbf{x}$ to separate inequality constraint from equality constraint. Then (6.78) can be rewritten as:

$$\min_{\mathbf{x},\mathbf{z}} \sum_{t=1}^{T} f_t(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{a}, \mathbf{B}\mathbf{x} = \mathbf{z} \ , \tag{6.79}$$

where $g(\mathbf{z})$ is the indicator function of box constraint $\mathbf{z} \leq \mathbf{b}$. The augmented Lagrangian for (6.79) is as follows:

$$\begin{aligned} L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}, \mathbf{v}) = & f_t(\mathbf{x}) + g(\mathbf{z}) + \langle \mathbf{u}, \mathbf{A}\mathbf{x} - \mathbf{a} \rangle + \langle \mathbf{v}, \mathbf{B}\mathbf{x} - \mathbf{z} \rangle \\ & + \frac{\rho_{\mathbf{u}}}{2} \|\mathbf{A}\mathbf{x} - \mathbf{a}\|_2^2 + \frac{\rho_{\mathbf{v}}}{2} \|\mathbf{B}\mathbf{x} - \mathbf{z}\|_2^2 \ , \end{aligned} \tag{6.80}$$

where $\mathbf{u}, \mathbf{v}$ are dual variables and the penalty parameters $\rho_{\mathbf{u}}, \rho_{\mathbf{v}} > 0$. Let the Bregman divergence in the $\mathbf{x}$ update in (6.6) be the quadratic function. We have the following OADM updates for (6.79):

$$\begin{aligned} \mathbf{x}_{t+1} = \operatorname*{argmin}_{\mathbf{x}} \Big\{ & f_t(\mathbf{x}) + \langle \mathbf{u}_t, \mathbf{A}\mathbf{x} - \mathbf{a} \rangle + \langle \mathbf{v}_t, \mathbf{B}\mathbf{x} - \mathbf{z}_t \rangle + \frac{\rho_{\mathbf{u}}}{2} \|\mathbf{A}\mathbf{x} - \mathbf{a}\|_2^2 \\ & + \frac{\rho_{\mathbf{v}}}{2} \|\mathbf{B}\mathbf{x} - \mathbf{z}_t\|_2^2 + \frac{\eta}{2} \|\mathbf{x} - \mathbf{x}_t\|_2^2 \Big\} \ , \end{aligned} \tag{6.81}$$

$$\mathbf{z}_{t+1} = \operatorname*{argmin}_{\mathbf{z} \leq \mathbf{b}} \ \Big\{ \langle \mathbf{v}_t, \mathbf{B}\mathbf{x}_{t+1} - \mathbf{z} \rangle + \frac{\rho_{\mathbf{v}}}{2} \|\mathbf{B}\mathbf{x}_{t+1} - \mathbf{z}\|_2^2 \Big\} \ , \tag{6.82}$$

$$\mathbf{u}_{t+1} = \mathbf{u}_t + \rho_{\mathbf{u}}(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{a}) \ , \tag{6.83}$$

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \rho_{\mathbf{v}}(\mathbf{B}\mathbf{x}_{t+1} - \mathbf{z}_{t+1}) \ , \tag{6.84}$$

where $\eta \geq 0$. The $\mathbf{x}$-update has a closed-form solution when f_t is a linear or quadratic functions, or the ℓ_1 norm. If the $\mathbf{x}$-update does not have a closed-form solution, we can linearize f_t at $\mathbf{x}_t$ as in Section 6.5.1, which leads to a closed-form solution. Further, the z-update has a closed-form solution of the following form:

$$\mathbf{z}_{t+1} = \min\{B\mathbf{x}_{t+1} + \mathbf{y}_t/\rho, \mathbf{b}\} . \tag{6.85}$$

Thus, OADM gives a projection-free online algorithm for optimization problems under linear constraints, e.g., linear and quadratic programming. In contrast, state-of-the-art online learning algorithms require the projection onto the constraints at each round, which amounts to solving a linear or quadratic program [77].

6.6 Experimental Results

In this section, we use OADM to solve generalized lasso problems [19], including lasso [189] and total variation (TV) problem [169]. We present simulation results to show the convergence of the objective as well as constraints in OADM. We also compare it with batch ADM and two other online learning algorithms: FOBOS [53] and regularized dual averaging (RDA) [210] in selecting sparse dimension in lasso and recovering data in total variation.

6.6.1 Generalized Lasso

The generalized lasso problem is formulated as follows:

$$\min_{\mathbf{x}} \frac{1}{N} \sum_{t=1}^{N} \|\mathbf{a}_t \mathbf{x} - b_t\|_2^2 + \lambda |\mathbf{D}\mathbf{x}|_1 , \tag{6.86}$$

where $\mathbf{a}_t \in \mathbb{R}^{1\times n}, \mathbf{x} \in \mathbb{R}^{n\times 1}, \mathbf{D} \in \mathbb{R}^{m\times n}$ and b_t is a scalar. If $\mathbf{D} = \mathbf{I}$, (6.86) yields the lasso. If $\mathbf{D}$ is an upper bidiagonal matrix with diagonal 1 and off-diagonal -1, (6.86) becomes the problem of total variation. The ADM form of (6.86) is:

$$\min_{\mathbf{D}\mathbf{x}=\mathbf{z}} \frac{1}{N} \sum_{t=1}^{N} \|\mathbf{a}_t \mathbf{x} - b_t\|_2^2 + \lambda |\mathbf{z}|_1 , \tag{6.87}$$

where $\mathbf{z} \in \mathbb{R}^{m\times 1}$. The augmented Lagrangian at round t is

$$L_\rho = \|\mathbf{a}_t \mathbf{x} - b_t\|_2^2 + \lambda|\mathbf{z}|_1 + \langle \mathbf{y}, \mathbf{D}\mathbf{x} - \mathbf{z}\rangle + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z}\|_2^2 .$$

The three updates of OADM yield the following closed-form updates:

$$\mathbf{x}_{t+1} = (\mathbf{a}_t^T\mathbf{a}_t + \rho\mathbf{D}^T\mathbf{D} + \eta)^{-1}\mathbf{v}_t \ , \tag{6.88}$$

$$\mathbf{z}_{t+1} = S_{\lambda/\rho}(\mathbf{D}\mathbf{x}_{t+1} + \mathbf{u}_t) \ , \tag{6.89}$$

$$\mathbf{u}_{t+1} = \mathbf{u}_t + \mathbf{D}\mathbf{x}_{t+1} - \mathbf{z}_{t+1} \ , \tag{6.90}$$

where $\mathbf{u} = \mathbf{y}/\rho$, $\mathbf{v}_t = \mathbf{a}_t^T b_t + \rho\mathbf{D}^T(\mathbf{z}_t - \mathbf{u}_t) + \eta\mathbf{x}_t$, and $S_{\lambda/\rho}$ denotes the soft thresholding operator or a shrinkage operator defined as

$$S_{\lambda/\rho}(k) = \begin{cases} k - \lambda/\rho, & k > \lambda/\rho \\ 0, & |\mathbf{x}| \leq \lambda/\rho \\ k + \lambda/\rho, & k < -\lambda/\rho \end{cases} \ , \tag{6.91}$$

which is a simple element-wise operation.

For lasso, the $\mathbf{x}$-update is

$$\mathbf{x}_{t+1} = (\mathbf{v}_t - (\eta + \rho + \mathbf{a}_t\mathbf{a}_t^T)^{-1}\mathbf{a}_t^T(\mathbf{a}_t\mathbf{v}_t))/(\eta + \rho) \ ,$$

where the inverse term is a scalar. The multiplication terms take $O(n)$ flops [71]. Thus, the $\mathbf{x}$-update can be done in $O(n)$ flops.

For total variation, we set $\eta = 0$ so that

$$\mathbf{x}_{t+1} = (\mathbf{Q}\mathbf{v}_t - (\rho + \mathbf{a}_t\mathbf{Q}\mathbf{a}_t^T)^{-1}\mathbf{Q}\mathbf{a}_t^T(\mathbf{a}_t\mathbf{Q}\mathbf{v}_t))/\rho \ ,$$

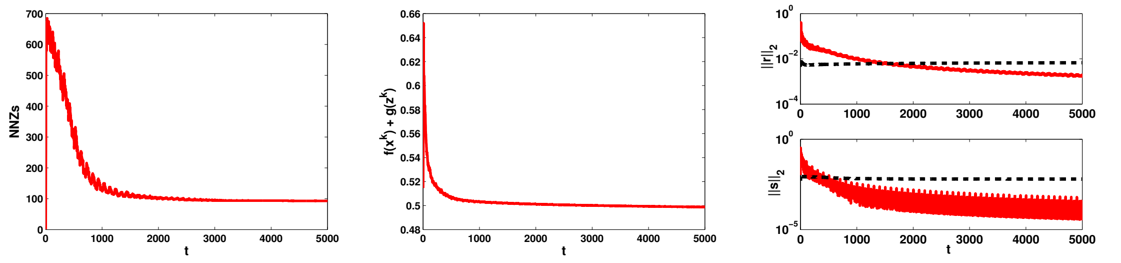
where $\mathbf{Q} = (\mathbf{D}^T\mathbf{D})^{-1}$. Since $\mathbf{D}$ is a bidiagonal matrix, $\mathbf{Q}\mathbf{v}_t$ and $\mathbf{Q}\mathbf{a}_t$ can be done in $O(n)$ flops [71, 19]. The inverse term is scalar and other multiplication terms cost $O(n)$ flops. Overall, the $\mathbf{x}$-update can be carried out in $O(n)$ flops.

In both cases, the three updates (6.88)-(6.90) can be done in $O(n)$ flops. In contrast, in batch ADM, the complexity of $\mathbf{x}$-update could be as high as $O(n^3)$ or $O(n^2)$ by caching factorizations [19].

FOBOS and RDA cannot directly solve the TV term. We first reformulate the total variation in the lasso form such that

$$\min_{\mathbf{y}} \frac{1}{N}\sum_{t=1}^{N} \|\mathbf{a}_t\mathbf{D}^{-1}\mathbf{y} - \mathbf{b}\|_2^2 + \lambda|\mathbf{y}|_1 \ , \tag{6.92}$$

where $\mathbf{y} = \mathbf{D}\mathbf{x}$. FOBOS and RDA can solve the above lasso problem and get $\mathbf{y}$. $\mathbf{x}$ can be recovered by using $\mathbf{x} = \mathbf{D}^{-1}\mathbf{y}$.



(a) Sparsity.

(b) Objective.

(c) Constraints (top), primal residual (bottom).

Figure 6.1: The convergence of sparsity, objective value and constraints for lasso in OADM with $q = 0.5, \rho = 1, \eta = t$.

6.6.2 Simulation

Our experiments mainly follow the lasso and total variation examples in [19],[2] although we modified the code to accommodate our setup. We first randomly generated $\mathbf{A}$ with N examples of dimensionality n. $\mathbf{A}$ is then normalized along the columns. Then, a true $\mathbf{x}_0$ is randomly generated with certain sparsity pattern for lasso and TV. For lasso, we set the number of nonzeros (NNZs) k in $\mathbf{x}_0$ as 100, i.e., $k = 100$. For TV, we first set $\mathbf{x}_0$ to be a vector of ones, then randomly select some blocks of random size in $\mathbf{x}_0$ and reset their value to a random value from $[1, 10]$. $\mathbf{b}$ is calculated by adding Gaussian noise to $\mathbf{A}\mathbf{x}_0/N$. In all experiments, $N = 100$, which facilitates the matrix inverse in ADM. For lasso, we try different combination of parameters from $n = [1000, 5000]$, $\rho = [0.1, 1, 10]$ and $q = [0.1, 0.5]$ for $\lambda = q \times |\mathbf{A}^T b/N|_\infty$. All experiments are implemented in Matlab.

Convergence: We go through the examples 100 times using OADM. Figure 6.1(a) shows that NNZs converge to a value close to the actual $k = 100$ before $t = 2000$. Figure 6.1(b) shows the convergence of objective value. In Figure 6.1(c), the dashed lines are the standard stopping criteria used in ADM [19]. Figure 6.1(c) shows that the equality constraint (top) and primal residual (bottom) are satisfied in the online setting. While the objective converges fast, the equality constraints take relatively more time to be satisfied.

Sparsity: We compare NNZs found by batch ADM and three online learning algorithms,

[2] `http://www.stanford.edu/~boyd/papers/admm/`

including OADM, FOBOS, and RDA. We set $\eta = 1000$ for OADM and $\gamma = 1$ for RDA. For FOBOS, we use a time varying parameter $\rho_t = \rho/\sqrt{t}$. For online learning algorithms, we go through the examples 100 times. We run the experiment 20 times and the average results are plotted. We show the results for $q = 0.5$ in Figure 2, where n is 1000 for the first three figures (a)-(c) and 5000 for the last three. While ADM and RDA tend to give the sparsest results, OADM seems more conservative and converges to reasonably sparse solutions. Figure 2 shows OADM is closest to the actual NNZs 100. The NNZs in FOBOS is large and oscillates in a big range, which has also been observed in [210].

Total Variation: We compare the patterns found by the four algorithms. For all algorithms, $N = 100, n = 1000, \lambda = 0.001$ and ρ is chosen through cross validation. In RDA, $\gamma = 100$. Recall that $\eta = 0$ in OADM. While we use a fixed ρ for OADM and RDA, FOBOS uses $\rho_t = \rho/\sqrt{t}$. Figure 6.3 shows the three different patterns and results found by the algorithms. ADM seems to follow the pattern with oscillation. OADM is smoother and generally follows the trend of the patterns. For the first two examples, FOBOS works well and the patterns found by RDA tend to be flat. In the last example, both FOBOS and RDA oscillate.

Appendix

6.A Proof of Theorem 19 and 21 in Case 2 in Section 6.5.1

Proof of Theorem 19 The first order derivative is 0, i.e.,

$$f'_t(\mathbf{x}_t) + \mathbf{A}^T\{\mathbf{y}_t + \rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_t - \mathbf{B}\mathbf{z}_t - \mathbf{c})\} + \eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t)) = 0 \ , \tag{6.93}$$

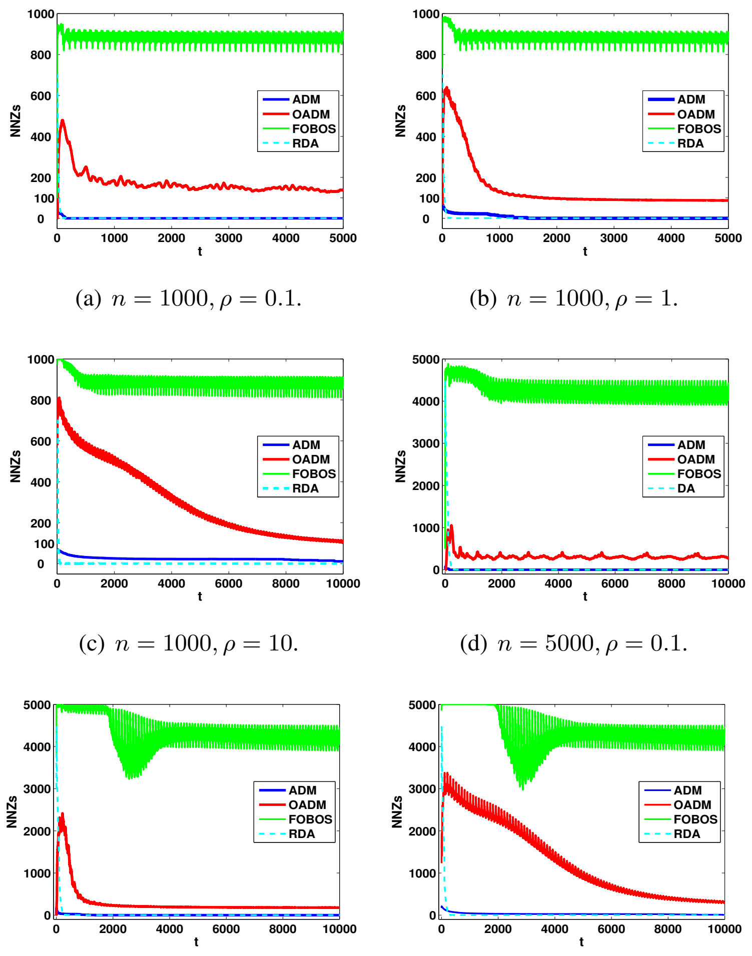
Rearranging the terms yields

$$-\mathbf{A}^T(\mathbf{y}_t + \rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t - \mathbf{c})) - \eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t)) = f'_t(\mathbf{x}_t) \ , \tag{6.94}$$

where the left hand side is same as (6.15). Therefore, $\langle f'_t(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*)$ can be written as the right hand side of (6.16). Using the convexity of f_t, we have

$$\begin{aligned} f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) &\le \langle f'_t(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^*\rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) \\ &= \langle f'_t(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) + \langle f'_t(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}_{t+1}\rangle \ . \end{aligned} \tag{6.95}$$

Applying (6.18) for the last term, we have (6.19). Therefore, Theorem 19 holds for Case 2.



(a) $n = 1000, \rho = 0.1$.

(b) $n = 1000, \rho = 1$.

(c) $n = 1000, \rho = 10$.

(d) $n = 5000, \rho = 0.1$.

(e) $n = 5000, \rho = 1$.

(f) $n = 5000, \rho = 10$.

Figure 6.2: The NNZs found by OADM, ADM, FOBOS and RDA with $q = 0.5$ for lasso. OADM is closest to the actual NNZs.

Proof of Theorem 21 Using the strong convexity of f_t and g defined in (6.25) and (6.26) respectively, we have

$$
\begin{aligned}
& f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) \\
& \leq \langle f_t'(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_t) + \langle g'(\mathbf{z}_{t+1}), \mathbf{z}_{t+1} - \mathbf{z}^* \rangle - \frac{\beta_2}{2} \|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 \\
& = \langle f_t'(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}^* \rangle + \langle f_t'(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}_{t+1} \rangle + \langle g'(\mathbf{z}_{t+1}), \mathbf{z}_{t+1} - \mathbf{z}^* \rangle
\end{aligned}
$$

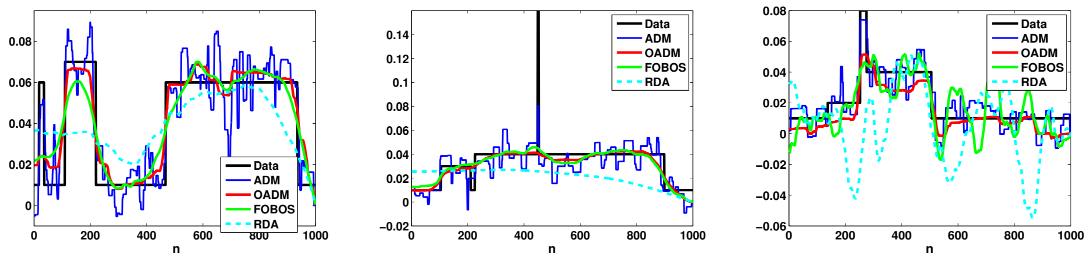


Figure 6.3: The TV patterns found by OADM, ADM, FOBOS and RDA. OADM is the best in recovering the patterns.

$$- \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_t) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 \, . \tag{6.96}$$

The first four terms are the same as in (6.95), which can be reduced to (6.19). Therefore, adding the last two terms to (6.19), we have

$$\begin{aligned} & f_t(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) \\ \leq & \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \\ & + \frac{1}{2\alpha\eta}\|f_t'(\mathbf{x}_t)\|_q^2 + \eta(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_t) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2 \, . \end{aligned} \tag{6.97}$$

Summing over t from 1 to T, we have

$$\begin{aligned} R_1(T) \leq & \sum_{t=1}^{T} \frac{1}{2\rho_{t+1}}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{1}{2\beta}\sum_{t=0}^{T}\frac{1}{\eta_{t+1}}\|f_t'(\mathbf{x}_t)\|_2^2 \\ & + \sum_{t=1}^{T}(\frac{\rho_{t+1}}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) - \frac{\beta_2}{2}\|\mathbf{z}^* - \mathbf{z}_{t+1}\|_2^2) \\ & + \sum_{t=1}^{T}(\eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_t)) \, . \end{aligned} \tag{6.98}$$

The difference between (6.98) and (6.32) lies in the last term. Setting $\eta_t = \beta_1 t$, we have the following telescoping sum for the last term :

$$\sum_{t=1}^{T}(\eta_{t+1}(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) - \beta_1 B_\phi(\mathbf{x}^*, \mathbf{x}_t))$$

$$\leq \eta_2 B_\phi(\mathbf{x}^*, \mathbf{x}_1) + \sum_{t=2}^{T} B_\phi(\mathbf{x}^*, \mathbf{x}_t)(\eta_{t+1} - \eta_t - \beta_1)$$
$$= 2\beta_1 D_{\mathbf{x}}^2 \,, \tag{6.99}$$

which is the same as (6.35). Therefore, Theorem 21 holds for the Case 2.

6.B Proof of Stochastic Convergence Rates

Although the proof is based on Case 2 in Section 6.1, Case 3 and 4 will follow automatically. In the stochastic setting, replacing $f_t'(\mathbf{x}_t)$ by $f'(\mathbf{x}_t, \xi_t)$ in (6.94) gives

$$-\mathbf{A}^T(\mathbf{y}_t + \rho\mathbf{A}^T(\mathbf{A}\mathbf{x}_{t+1} - \mathbf{B}\mathbf{z}_t - \mathbf{c})) - \eta(\nabla\phi(\mathbf{x}_{t+1}) - \nabla\phi(\mathbf{x}_t)) = f'(\mathbf{x}_t, \xi_t) \,, \tag{6.100}$$

(a) Replacing $f_t(\mathbf{x}_t), f_t'(\mathbf{x}_t)$ by $f(\mathbf{x}_t), f'(\mathbf{x}_t, \xi_t)$ respectively in (6.95) gives

$$\begin{aligned} & f(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_t - \mathbf{x}^* \rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) \\ & = \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_{t+1} - \mathbf{x}^* \rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) + \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_t - \mathbf{x}_{t+1} \rangle \,. \end{aligned} \tag{6.101}$$

As a result, we have the following result by replacing $f_t(\mathbf{x}_t), f_t'(\mathbf{x}_t)$ by $f(\mathbf{x}_t), f'(\mathbf{x}_t, \xi_t)$ in (6.19)

$$\begin{aligned} & f(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \\ & \leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) - \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2 \\ & \quad + \frac{1}{2\alpha\eta}\|f'(\mathbf{x}_t, \xi_t)\|_q^2 + \eta(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) \,. \end{aligned} \tag{6.102}$$

Moving the term $\frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_t - \mathbf{c}\|_2^2$ to the left hand side and using Lemma 9, we have

$$\begin{aligned} & f(\mathbf{x}_t) + g(\mathbf{z}_{t+1}) - (f_t(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \\ & \leq \frac{1}{2\rho}(\|\mathbf{y}_t\|_2^2 - \|\mathbf{y}_{t+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_t\|_2^2 - \|\mathbf{B}\mathbf{z}^* - \mathbf{B}\mathbf{z}_{t+1}\|_2^2) \\ & \quad + \frac{1}{2\alpha\eta}\|f'(\mathbf{x}_t, \xi_t)\|_q^2 + \eta(B_\phi(\mathbf{x}^*, \mathbf{x}_t) - B_\phi(\mathbf{x}^*, \mathbf{x}_{t+1})) \,. \end{aligned} \tag{6.103}$$

Summing over t from 0 to $T-1$ and following the derivation in (6.20), we have

$$\sum_{t=1}^{T} \left[f(\mathbf{x}_t) + g(\mathbf{z}_t) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2}\|\mathbf{A}\mathbf{x}_{t+1} + \mathbf{B}\mathbf{z}_{t+1} - \mathbf{c}\|_2^2 \right]$$

$$\leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2} + \eta D_{\mathbf{x}}^2 + \frac{\|f'(\mathbf{x}_t, \xi_t)\|_q^2 T}{2\alpha\eta} \ . \tag{6.104}$$

Dividing both sides by T, applying the Jensen's inequality, we have

$$\begin{aligned} & f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2}\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2 \\ & \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2T} + \frac{\eta D_{\mathbf{x}}^2}{T} + \frac{\|f'(\mathbf{x}_t, \xi_t)\|_q^2}{2\alpha\eta} \ . \end{aligned} \tag{6.105}$$

Assume $\mathbf{E}[\|f'(\mathbf{x}_t, \xi_t)\|_q^2] \leq G_f^2$. Setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$ and taking expectation, we have

$$\mathbf{E}\left[f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2}\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2\right] \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2T} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{\alpha}\sqrt{T}} \ . \tag{6.106}$$

(6.66) follows by setting $\rho = \sqrt{T}$.

Assume $f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \geq -F$. Dividing both sides by $\frac{\rho}{2}$ and rearranging the terms yield

$$\mathbf{E}\left[\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2\right] \leq \frac{2F}{\rho} + \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{T} + \frac{2\sqrt{2} G_f D_{\mathbf{x}}}{\rho\sqrt{\alpha}\sqrt{T}} \ . \tag{6.107}$$

Setting $\rho = \sqrt{T}$ gives (6.67).

(b) Using the convexity of f, we have

$$f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \langle f'(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^*\rangle = \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_{t+1} - \mathbf{x}^*\rangle + \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_t - \mathbf{x}_{t+1}\rangle + \epsilon_t \ . \tag{6.108}$$

where

$$\epsilon_t = \langle f'(\mathbf{x}_t) - f'(\mathbf{x}_t, \xi_t), \mathbf{x}_t - \mathbf{x}^*\rangle \ . \tag{6.109}$$

Let $\mathcal{F}$ be a filtration with $\xi_t \in \mathcal{F}_t$ for $t \leq T$. Since $\mathbf{x}_t \in \mathcal{F}_{t-1}$,

$$\mathbf{E}[\epsilon_t|\mathcal{F}_{t-1}] = \langle f'(\mathbf{x}_t) - \mathbf{E}[f'(\mathbf{x}_t, \xi_t)|\mathcal{F}_{t-1}], \mathbf{x}_t - \mathbf{x}^*\rangle = 0 \ . \tag{6.110}$$

Therefore, $\sum_{t=1}^T \epsilon_t$ is a martingale difference sequence. Assuming $B_\phi(\mathbf{x}^*, \mathbf{x}^t) \leq D_{\mathbf{x}}^2$, $\|\mathbf{x}_t - \mathbf{x}^*\|_p \leq \sqrt{\frac{2}{\alpha}} D_{\mathbf{x}}$. We have

$$|\epsilon_t| \leq \|f'(\mathbf{x}_t) - f'(\mathbf{x}_t, \xi_t)\|_q \|\mathbf{x}_t - \mathbf{x}^*\|_p \leq 2\sqrt{\frac{2}{\alpha}} D_{\mathbf{x}} G_f \ . \tag{6.111}$$

Applying Azuma-Hoeffding inequality [3] on $\sum_{t=1}^{T} \epsilon_t$ yields

$$P(\sum_{t=1}^{T} \epsilon_t \geq \varepsilon) \leq \exp\left(-\frac{\alpha \varepsilon^2}{16 T D_{\mathbf{x}}^2 G_f^2}\right) \ . \tag{6.112}$$

Combing (6.101) and (6.108), we have

$$\begin{aligned} f(\mathbf{x}_t) + & g(\mathbf{z}_{t+1}) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \leq \langle f_t'(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) \\ & = \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_{t+1} - \mathbf{x}^* \rangle + g(\mathbf{z}_{t+1}) - g(\mathbf{z}^*) + \langle f'(\mathbf{x}_t, \xi_t), \mathbf{x}_t - \mathbf{x}_{t+1} \rangle + \epsilon_t \ . \end{aligned} \tag{6.113}$$

As a result, (6.105) becomes

$$\begin{aligned} & f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2} \|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2 \\ & \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2 \rho}{2T} + \frac{\eta D_{\mathbf{x}}^2}{T} + \frac{\|f'(\mathbf{x}_t, \xi_t)\|_q^2}{2\alpha\eta} + \frac{1}{T}\sum_{t=1}^{T} \epsilon_t \ . \end{aligned} \tag{6.114}$$

Assuming $\|f'(\mathbf{x}_t, \xi_t)\|_q \leq G_f$ and setting $\eta = \frac{G_f\sqrt{T}}{D_{\mathbf{x}}\sqrt{2\alpha}}$, $\rho = \sqrt{T}$, we have

$$\begin{aligned} & f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) + \frac{\rho}{2} \|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2 \\ & \leq \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{2\sqrt{T}} + \frac{\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{\alpha}\sqrt{T}} + \frac{1}{T}\sum_{t=1}^{T} \epsilon_t \ . \end{aligned} \tag{6.115}$$

Applying (6.112) gives (6.68).

Assume $f(\bar{\mathbf{x}}_T) + g(\bar{\mathbf{z}}_T) - (f(\mathbf{x}^*) + g(\mathbf{z}^*)) \geq -F$. In (6.115), dividing both sides by $\frac{\rho}{2} = \frac{\sqrt{T}}{2}$ and rearranging the terms yield

$$\|\mathbf{A}\bar{\mathbf{x}}_T + \mathbf{B}\bar{\mathbf{z}}_T + \mathbf{c}\|_2^2 \leq \frac{2F}{\rho} + \frac{\lambda_{\max}^{\mathbf{B}} D_{\mathbf{z}}^2}{T} + \frac{2\sqrt{2} G_f D_{\mathbf{x}}}{\sqrt{\alpha} T} + \frac{1}{T}\sum_{t=1}^{T} \epsilon_t \ . \tag{6.116}$$

Applying (6.112) yields (6.69). ■

Part III

Applications

Chapter 7

Bethe-ADMM for Tree Decomposition based Parallel MAP Inference

7.1 Introduction

Given a discrete graphical model with known structure and parameters, the problem of finding the most likely configuration of the states is known as the *maximum a posteriori* (MAP) inference problem [199]. Existing approaches to solving MAP inference problems on graphs with cycles often consider a graph-based linear programming (LP) relaxation of the integer program [31, 158, 197] .

To solve the graph-based LP relaxation problem, two main classes of algorithms have been proposed. The first class of algorithms are dual LP algorithms [68, 94, 104, 179, 180, 187], which uses the dual decomposition and solves the dual problem. The two main approaches to solving the dual problems are block coordinate descent [68] and sub-gradient algorithms [104]. The coordinate descent algorithms are empirically faster, however, they may not reach the dual optimum since the dual problem is not strictly convex. Recent advances in coordinate descent algorithms perform tree-block updates [180, 187]. The sub-gradient methods, which are guaranteed to converge to the global optimum, can be slow in practice. For a detailed discussion on dual MAP algorithms, we refer the readers to [179]. The second class of algorithms are primal LP algorithms like the proximal algorithm [158]. The advantage of such algorithms is that it can choose different Bregman divergences as proximal functions which can take the graph structure into account. However, the proximal algorithms do not have a closed form update at each iteration in general and thus lead to double-loop algorithms.

As solving MAP inference in large scale graphical models is becoming increasingly important, in recent work, parallel MAP inference algorithms [133, 139] based on the alternating direction method of multipliers (ADMM) [19] have been proposed. As a primal-dual algorithm, ADMM combines the advantage of dual decomposition and the method of multipliers, which is guaranteed to converge globally and at a rate of $O(1/T)$ even for non-smooth problems [200]. ADMM has also been successfully used to solve large scale problem in a distributed manner [19].

Design of efficient parallel algorithms based on ADMM by problem decomposition has to consider a key tradeoff between the number of subproblems and the size of each subproblem. Having several simple subproblems makes solving each problem easy, but one has to maintain numerous dual variables to achieve consensus. On the other hand, having a few subproblems makes the number of constraints small, but each subproblem needs an elaborate often iterative algorithm, yielding a double-loop. Existing ADMM based algorithms for MAP inference [133, 139] decompose the problem into several simple subproblems, often based on single edges or local factors, so that the subproblems are easy to solve. However, to enforce consensus among the shared variables, such methods have to use dual variables proportional to the number of edges or local factors, which can make convergence slow on large graphs.

To overcome the limitations of existing ADMM methods for MAP inference, we propose a novel parallel algorithm based on tree decomposition. The individual trees need not be spanning and thus includes both edge decomposition and spanning tree decomposition as special cases. Compared to edge decomposition, tree decomposition has the flexibility of increasing the size of subproblems and reducing the number of subproblems by considering the graph structure. Compared to the tree block coordinate descent [180], which works with one tree at a time, our algorithm updates all trees in parallel. Note that the tree block coordinate descent algorithm in [187] updates disjoint trees within a forest in parallel, whereas our updates consider overlapping trees in parallel.

However, tree decomposition raises a new problem: the subproblems cannot be solved efficiently in the ADMM framework and requires an iterative algorithm, yielding a double-loop algorithm [133, 158]. To efficiently solve the subproblem on a tree, we propose a novel inexact ADMM algorithm called Bethe-ADMM, which uses a Bregman divergence induced by Bethe entropy on a tree, instead of the standard quadratic divergence, as the proximal function. The resulting subproblems on each tree can be solved exactly in linear time using the sum-product algorithm [107]. However, the proof of convergence for the standard ADMM does not apply to Bethe-ADMM. We prove global convergence of Bethe-ADMM and establish a $O(1/T)$ convergence rate, which is the same as the standard ADMM [200]. Overall, Bethe-ADMM overcomes the limitations of existing ADMM based MAP inference algorithms [133, 139] and provides the flexibility required in designing efficient parallel algorithm through: (i) Tree decomposition, which can take the graph structure into account and greatly reduce the number of variables participating in the consensus and (ii) the Bethe-ADMM algorithm, which yields efficient updates for each subproblem.

We compare the performance of Bethe-ADMM with existing methods on both synthetic and real datasets and illustrate four aspects. First, Bethe-ADMM is faster than existing primal LP methods in terms of convergence. Second, Bethe-ADMM is competitive with existing dual methods in terms of quality of solutions obtained. Third, in certain graphs, tree decomposition leads to faster convergence than edge decomposition for Bethe-ADMM. Forth, parallel Bethe-ADMM, based on Open MPI, gets substantial speed-ups over sequential Bethe-ADMM. In particular, we show almost linear speed-ups with increasing number of cores on a graph with several million nodes.

The rest of this chapter is organized as follows: We review the MAP inference problem in Section 7.2. In Section 7.3, we introduce the Bethe-ADMM algorithm and prove its global convergence. We discuss empirical evaluation in Section 7.4.

7.2 Background and Related Work

We first introduce some basic background on Markov Random Fields (MRFs). Then we briefly review existing ADMM based MAP inference algorithms in the literature. We mainly focus on pairwise MRFs and the discussions can be easily carried over to MRFs with general factors.

7.2.1 Problem Definition

A pairwise MRF is defined on an undirected graph $G = (V, E)$, where V is the vertex set and E is the edge set. Each node $u \in V$ has a random variable X_u associated with it, which can take value x_u in some discrete space $\mathcal{X} = \{1, \ldots, k\}$. Concatenating all the random variables $X_u, \forall u \in V$, we obtain an n dimensional random vector $\boldsymbol{X} = \{X_u | u \in V\} \in \mathcal{X}^n$. We assume that the distribution P of $\boldsymbol{X}$ is a Markov Random Field [199], meaning that it factors according to the structure of the undirected graph G as follows: With $f_u : \mathcal{X} \mapsto \mathbb{R}, \forall u \in V$ and $f_{uv} : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}, \forall (u,v) \in E$ denoting nodewise and edgewise potential functions respectively, the distribution takes the form $P(\mathbf{x}) \propto \exp\left\{\sum_{u\in V} f_u(x_u) + \sum_{(u,v)\in E} f_{uv}(x_u, x_v)\right\}$.

An important problem in the context of MRF is that of *maximum a posteriori* (MAP) inference, which is the following integer programming (IP) problem:

$$\mathbf{x}^* \in \operatorname{argmax}_{\mathbf{x}\in\mathcal{X}^n} \left\{ \sum_{u\in V} f_u(x_u) + \sum_{(u,v)\in E} f_{uv}(x_u, x_v) \right\} . \tag{7.1}$$

The complexity of (7.1) depends critically on the structure of the underlying graph. When G is a tree structured graph, the MAP inference problem can be solved efficiently via the max-product algorithm [107]. However, for an arbitrary graph G, the MAP inference algorithm is usually computationally intractable. The intractability motivates the development of algorithms to solve the MAP inference problem approximately. In this chapter, we focus on the linear programming (LP) relaxation method [31, 197]. The LP relaxation of MAP inference problem is defined on a set of pseudomarginals μ_u and μ_{uv}, which are non-negative, normalized and locally consistent [31, 197]:

$$\begin{aligned} \mu_u(x_u) \geq 0\,, &\quad \forall u \in V\,, \\ \sum_{x_u\in\mathcal{X}_u} \mu_u(x_u) = 1, &\quad \forall u \in V\,, \\ \mu_{uv}(x_u, x_v) \geq 0, &\quad \forall (u,v) \in E\,, \\ \sum_{x_u\in\mathcal{X}_u} \mu_{uv}(x_u, x_v) = \mu_v(x_v), &\quad \forall (u,v) \in E\,. \end{aligned} \tag{7.2}$$

We denote the polytope defined by (7.2) as $L(G)$. The LP relaxation of MAP inference problem (7.1) becomes solving the following LP:

$$\max_{\boldsymbol{\mu}\in L(G)} \langle \boldsymbol{\mu}, \boldsymbol{f} \rangle\,. \tag{7.3}$$

If the solution $\boldsymbol{\mu}$ to (7.3) is an integer solution, it is guaranteed to be the optimal solution of (7.1). Otherwise, one can apply rounding schemes [156, 158] to round the fractional solution to an integer solution.

7.2.2 ADMM based MAP Inference Algorithms

In recent years, ADMM [133, 139] has been used to solve large scale MAP inference problems. To solve (7.3) using ADMM, we need to split nodes or/and edges and introduce equality constraints to enforce consensus among the shared variables. The algorithm in [133] adopts edge decomposition and introduces equality constraints for shared nodes. Let d_i be the degree of node i. The number of equality constraints in [133] is $O(\sum_{i=1}^{|V|} d_i k)$, which is approximately equal to $O(|E|k)$. For binary pairwise MRFs, the subproblems for the ADMM in [133] have closed-form solutions. For multi-valued MRFs, however, one has to first binarize the MRFs which introduces additional $|V|k$ variables for nodes and $2|E|k^2$ variables for edges. The binarization process increases the number of factors to $O(|V| + 2|E|k)$ and the complexity of solving each subproblem increases to $O(|E|k^2 \log k)$. We note that in a recent work [132], the active set method is employed to solve the quadratic problem for arbitrary factors. A generalized variant of [133] which does not require binarization is presented in [139]. We refer to this algorithm as Primal ADMM and use it as a baseline in Section 7.4. Although each subproblem in primal ADMM can be efficiently solved, the number of equality constraints and dual variables is $O(2|E|k + |E|k^2)$. In [139], ADMM is also used to solve the dual of (7.1). We refer to this algorithm as the Dual ADMM algorithm and use it as a baseline in Section 7.4. The dual ADMM works for multi-valued MRFs and has a linear time algorithm for each subproblem, but the number of equality constraint is $O(2|E|k + |E|k^2)$.

7.3 Algorithm and Analysis

We first show how to solve (7.3) using ADMM based on tree decomposition. The resulting algorithm can be a double-loop algorithm since some updates do not have closed form solutions. We then introduce the Bethe-ADMM algorithm where every subproblem can be solved exactly and efficiently, and analyze its convergence properties.

7.3.1 ADMM for MAP Inference

We first show how to decompose (7.3) into a series of subproblems. We can decompose the graph G into overlapping subgraphs and rewrite the optimization problem with consensus constraints to enforce the pseudomarginals on subgraphs (local variables) to agree with $\boldsymbol{\mu}$ (global variable). Throughout the chapter, we focus on tree-structured decompositions. To be more specific, let $\mathbb{T} = \{(V_1, E_1), \dots, (V_{|\mathbb{T}|}, E_{|\mathbb{T}|})\}$ be a collection of subgraphs of G which satisfies two criteria: (i) Each subgraph $\tau = (V_\tau, E_\tau)$ is a tree-structured graph and (ii) Each node $u \in V$ and each edge $(u, v) \in E$ is included in at least one subgraph $\tau \in \mathbb{T}$. We also introduce local variable $\boldsymbol{m}_\tau \in L(\tau)$ which is the pseudomarginal [31, 197] defined on each subgraph τ. We use $\boldsymbol{\theta}_\tau$ to denote the potentials on subgraph τ. We denote $\boldsymbol{\mu}_\tau$ as the components of global variable $\boldsymbol{\mu}$ that belong to subgraph τ. Note that since $\boldsymbol{\mu} \in L(G)$ and τ is a tree-structured subgraph of G, $\boldsymbol{\mu}_\tau$ always lies in $L(\tau)$. In the newly formulated optimization problem, we will impose consensus constraints for shared nodes and edges. For the ease of exposition, we simply use the equality constraint $\boldsymbol{\mu}_\tau = \boldsymbol{m}_\tau$ to enforce the consensus.

The new optimization problem we formulate based on graph decomposition is then as follows:

$$\min_{\boldsymbol{m}_\tau, \boldsymbol{\mu}} \quad \sum_{\tau=1}^{|\mathbb{T}|} \rho_\tau \langle \boldsymbol{m}_\tau, \boldsymbol{\theta}_\tau \rangle \tag{7.4}$$

$$\text{subject to} \quad \boldsymbol{m}_\tau - \boldsymbol{\mu}_\tau = 0, \quad \tau = 1, \dots, |\mathbb{T}| \tag{7.5}$$

$$\boldsymbol{m}_\tau \in L(\tau), \quad \tau = 1, \dots, |\mathbb{T}| \tag{7.6}$$

where ρ_τ is a positive constant associated with each subgraph. We use the consensus constraints (7.5) to make sure that the pseudomarginals agree with each other in the shared components across all the tree-structured subgraphs. Besides the consensus constraints, we also impose feasibility constraints (7.6), which guarantee that, for each subgraph, the local variable $\boldsymbol{m}_\tau$ lies in $L(\tau)$. When the constraints (7.5) and (7.6) are satisfied, the global variable $\boldsymbol{\mu}$ is guaranteed to lie in $L(G)$.

To make sure that problem (7.3) and (7.4)-(7.6) are equivalent, we also need to guarantee that

$$\min_{\boldsymbol{m}_\tau} \sum_{\tau=1}^{|\mathbb{T}|} \rho_\tau \langle \boldsymbol{m}_\tau, \boldsymbol{\theta}_\tau \rangle = \max_{\boldsymbol{\mu}} \langle \boldsymbol{\mu}, \boldsymbol{f} \rangle \ , \tag{7.7}$$

assuming the constraints (7.5) and (7.6) are satisfied. It is easy to verify that, as long as (7.7) is satisfied, the specific choice of ρ_τ and $\boldsymbol{\theta}_\tau$ do not change the problem. Let $\mathbf{1}[.]$ be a binary indicator function and $\boldsymbol{l} = -\boldsymbol{f}$. For any positive $\rho_\tau, \forall \tau \in \mathbb{T}$, e.g., $\rho_\tau = 1$, a simple approach to obtaining the potential $\boldsymbol{\theta}_\tau$ can be:

$$\theta_{\tau,u}(x_u) = \frac{l_u(x_u)}{\sum_{\tau'} \rho_{\tau'} \mathbf{1}[u \in V_{\tau'}]}, u \in V_\tau \ ,$$

$$\theta_{\tau,uv}(x_u,x_v) = \frac{l_{uv}(x_u,x_v)}{\sum_{\tau'} \rho_{\tau'} \mathbf{1}[(u,v) \in E_{\tau'}]}, (u,v) \in E(\tau) .$$

Let $\boldsymbol{\lambda}_\tau$ be the dual variable and $\beta > 0$ be the penalty parameter. The following updates constitute a single iteration of the ADMM [19]:

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \boldsymbol{m}_\tau, \rho_\tau \boldsymbol{\theta}_\tau + \boldsymbol{\lambda}_\tau^t \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau - \boldsymbol{\mu}_\tau^t||_2^2 , \tag{7.8}$$

$$\boldsymbol{\mu}^{t+1} = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{\tau=1}^{|\mathbb{T}|} \left(-\langle \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau^t \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau||_2^2 \right) , \tag{7.9}$$

$$\boldsymbol{\lambda}_\tau^{t+1} = \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}) . \tag{7.10}$$

In the tree based ADMM (7.8)-(7.10), the equality constraints are only required for shared nodes and edges. Assume there are m shared nodes and the shared node v_i has C_i^v copies and there are n shared edges and the shared edge e_j has C_j^e copies. The total number of equality constraints is $O(\sum_{i=1}^m C_i^v k + \sum_{j=1}^n C_j^e k^2)$. A special case of tree decomposition is edge decomposition, where only nodes are shared. In edge decomposition, $n = 0$ and the number of equality constraints is $O(\sum_{i=1}^m C_i^v k)$, which is approximately equal to $O(|E|k)$ and similar to [133]. In general, the number of shared nodes and edges in tree decomposition is much smaller than that in edge decomposition. The smaller number of equality constraints usually lead to faster convergence in achieving consensus. Now, the problem turns to whether the updates (7.8) and (7.9) can be solved efficiently, which we analyze below:

Updating $\boldsymbol{\mu}$: Since we have an unconstrained optimization problem (7.9) and the objective function decomposes component-wisely, taking the derivatives and setting them to zero yield the solution. In particular, let S_u be the set of subgraphs which contain node u, for the node components, we have:

$$\mu_u^{t+1}(x_u) = \frac{1}{|S_u|\beta} \sum_{\tau \in S_u} \left(\beta m_{\tau,u}^{t+1}(x_u) + \lambda_{\tau,u}^t(x_u) \right) . \tag{7.11}$$

(7.11) can be further simplified by observing that $\sum_{\tau \in S_u} \lambda_{\tau,u}^t(x_u) = 0$ [19]:

$$\mu_u^{t+1}(x_u) = \frac{1}{|S_u|} \sum_{\tau=1}^{T} m_{\tau,u}^{t+1}(x_u) . \tag{7.12}$$

Let S_{uv} be the subgraphs which contain edge (u,v). The update for the edge components can be similarly derived as:

$$\mu_{u,v}^{t+1}(x_u, x_v) = \frac{1}{|S_{uv}|} \sum_{\tau \in S_{uv}} m_{\tau,uv}^{t+1}(x_u, x_v) \ . \tag{7.13}$$

Updating $\boldsymbol{m}_\tau$: For (7.8), we need to solve a quadratic optimization problem for each tree-structured subgraph. Unfortunately, we do not have a close-form solution for (7.8) in general. One possible approach, similar to the proximal algorithm, is to first obtain the solution $\tilde{\boldsymbol{m}}_\tau$ to the unconstrained problem of (7.8) and then project $\tilde{\boldsymbol{m}}_\tau$ to $L(\tau)$:

$$\boldsymbol{m}_\tau = \text{argmin}_{\boldsymbol{m} \in L(\tau)} ||\boldsymbol{m} - \tilde{\boldsymbol{m}}_\tau||_2^2 \ . \tag{7.14}$$

If we adopt the cyclic Bregman projection algorithm [25] to solve (7.14), the algorithm becomes a double-loop algorithm, i.e., the cyclic projection algorithm projects the solution to each individual constraint of $L(\tau)$ until convergence and the projection algorithm itself is iterative. We refer to this algorithm as the Exact ADMM and use it as a baseline in Section 7.4.

7.3.2 Bethe-ADMM

Instead of solving (7.8) exactly, a common way in inexact ADMMs [97, 214] is to linearize the objective function in (7.8), i.e., the first order Taylor expansion at $\boldsymbol{m}_\tau^t$, and add a new quadratic penalty term such that

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\text{argmin}} \langle \mathbf{y}_\tau^t, \boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t \rangle + \frac{\alpha}{2} \|\boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t\|_2^2 \ , \tag{7.15}$$

where α is a positive constant and

$$\mathbf{y}_\tau^t = \rho_\tau \boldsymbol{\theta}_\tau + \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^t - \boldsymbol{\mu}_\tau^t) \ . \tag{7.16}$$

However, as discussed in the previous section, the quadratic problem (7.15) is generally difficult for a tree-structured graph and thus the conventional inexact ADMM does not lead to an efficient update for $\boldsymbol{m}_\tau$. By taking the tree structure into account, we propose an inexact minimization of (7.8) augmented with a Bregman divergence induced by the Bethe entropy. We show that the resulting proximal problem can be solved exactly and efficiently using the sum-product algorithm [107]. We prove that the global convergence of the Bethe-ADMM algorithm in Section 7.3.3.

The basic idea in the new algorithm is that we replace the quadratic term in (7.15) with a Bregman-divergence term $d_\phi(\boldsymbol{m}_\tau||\boldsymbol{m}_\tau^t)$ such that

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t, \boldsymbol{m}_\tau - \boldsymbol{m}_\tau^t \rangle + \alpha d_\phi(\boldsymbol{m}_\tau||\boldsymbol{m}_\tau^t) \ , \tag{7.17}$$

is efficient to solve for any tree τ. Expanding the Bregman divergence and removing the constants, we can rewrite (7.17) as

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t/\alpha - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{m}_\tau \rangle + \phi(\boldsymbol{m}_\tau). \tag{7.18}$$

For a tree-structured problem, what convex function $\phi(\boldsymbol{m}_\tau)$ should we choose? Recall that $\boldsymbol{m}_\tau$ defines the marginal distributions of a tree-structured distribution $p_{\boldsymbol{m}_\tau}$ over the nodes and edges:

$$\boldsymbol{m}_{\tau,u}(x_u) = \sum_{\neg x_u} p_{\boldsymbol{m}_\tau}(x_1, \ldots, x_u, \ldots, x_n), \ \forall u \in V_\tau \ ,$$
$$\boldsymbol{m}_{\tau,uv}(x_u, x_v) = \sum_{\neg x_u, \neg x_v} p_{\boldsymbol{m}_\tau}(x_1, \ldots, x_u, x_v, \ldots, x_n), \ \forall (uv) \in E_\tau .$$

It is well known that the sum-product algorithm [107] efficiently computes the marginal distributions for a tree structured graph. It can also be shown that the sum-product algorithm solves the following optimization problem [199] for tree τ for some constant $\boldsymbol{\eta}_\tau$:

$$\max_{\boldsymbol{m}_\tau \in L(\tau)} \langle \boldsymbol{m}_\tau, \boldsymbol{\eta}_\tau \rangle + H_{Bethe}(\boldsymbol{m}_\tau) \ , \tag{7.19}$$

where $H_{Bethe}(\boldsymbol{m}_\tau)$ is the Bethe entropy of $\boldsymbol{m}_\tau$ defined as:

$$H_{Bethe}(\boldsymbol{m}_\tau) = \sum_{u \in V_\tau} H_u(\boldsymbol{m}_{\tau,u}) - \sum_{(u,v) \in E_\tau} I_{uv}(\boldsymbol{m}_{\tau,uv}) \ , \tag{7.20}$$

where $H_u(\boldsymbol{m}_{\tau,u})$ is the entropy function on each node $u \in V_\tau$ and $I_{uv}(\boldsymbol{m}_{\tau,uv})$ is the mutual information on each edge $(u,v) \in E_\tau$.

Combing (7.18) and (7.19), we set $\boldsymbol{\eta}_\tau = \nabla\phi(\boldsymbol{m}_\tau^t) - \mathbf{y}_\tau^t/\alpha$ and choose ϕ to be the negative Bethe entropy of $\boldsymbol{m}_\tau$ so that (7.18) can be solved efficiently in linear time via the sum-product algorithm.

For the sake of completeness, we summarize the Bethe-ADMM algorithm as follows :

$$\boldsymbol{m}_\tau^{t+1} = \underset{\boldsymbol{m}_\tau \in L(\tau)}{\operatorname{argmin}} \langle \mathbf{y}_\tau^t/\alpha - \nabla\phi(\boldsymbol{m}_\tau^t), \boldsymbol{m}_\tau \rangle + \phi(\boldsymbol{m}_\tau) \ , \tag{7.21}$$

$$\boldsymbol{\mu}^{t+1} = \underset{\boldsymbol{\mu}}{\text{argmin}} \sum_{\tau=1}^{T} \left(-\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau \rangle + \frac{\beta}{2} ||\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau||_2^2 \right), \tag{7.22}$$

$$\boldsymbol{\lambda}_\tau^{t+1} = \boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}) \ , \tag{7.23}$$

where $\mathbf{y}_\tau^t$ is defined in (7.16) and $-\phi$ is defined in (7.20).

7.3.3 Convergence

We prove the global convergence of the Bethe-ADMM algorithm. We first bound the Bregman divergence d_ϕ:

Lemma 21 *Let $\boldsymbol{\mu}_\tau$ and $\boldsymbol{\nu}_\tau$ be two concatenated vectors of the pseudomarginals on a tree τ with n_τ nodes. Let $d_\phi(\boldsymbol{\mu}_\tau||\boldsymbol{\nu}_\tau)$ be the Bregman divergence induced by the negative Bethe entropy ϕ. Assuming $\alpha \geq \max_\tau\{\beta(2n_\tau - 1)^2\}$, we have*

$$\alpha d_\phi(\boldsymbol{\mu}_\tau||\boldsymbol{\nu}_\tau) \geq \frac{\beta}{2} \|\boldsymbol{\mu}_\tau - \boldsymbol{\nu}_\tau\|_2^2 \ . \tag{7.24}$$

Proof: Let $P_\tau(\mathbf{x})$ be a tree-structured distribution on a tree $\tau = (V_\tau, E_\tau)$, where $|V_\tau| = n_\tau$ and $|E_\tau| = n_\tau - 1$. The pseudomarginal $\boldsymbol{\mu}_\tau$ has a total of $2n_\tau - 1$ components, each being a marginal distribution. In particular, there are n_τ marginal distributions corresponding to each node $u \in V_\tau$, given by

$$\mu_{\tau,u}(x_u) = \sum_{\neg x_u} P_\tau(x_1, \dots, x_u, \dots, x_n) \ . \tag{7.25}$$

Thus, $\boldsymbol{\mu}_u$ is the marginal probability for node u.

Further, there are $n_\tau - 1$ marginal components corresponding to each edge $(u, v) \in E_\tau$, given by

$$\mu_{\tau,uv}(x_u, x_v) = \sum_{\neg(x_u,x_v)} P(x_1, \dots, x_u, \dots, x_v, \dots, x_n) \ . \tag{7.26}$$

Thus, $\boldsymbol{\mu}_{uv}$ is the marginal probability for nodes (u, v).

Let $\boldsymbol{\mu}_\tau, \boldsymbol{\nu}_\tau$ be two pseudomarginals defined on tree τ and $P_{\boldsymbol{\mu}_\tau}, P_{\boldsymbol{\nu}_\tau}$ be the corresponding tree-structured distributions. Making use of (7.25), we have

$$\|P_{\boldsymbol{\mu}_\tau} - P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_{\tau,u} - \boldsymbol{\nu}_{\tau,u}\|_1, \quad \forall u \in V_\tau \ . \tag{7.27}$$

Similarly, for each edge, we have the following inequality because of (7.26)

$$\|P_{\boldsymbol{\mu}_\tau}-P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_{\tau,uv}-\boldsymbol{\nu}_{\tau,uv}\|_1,\ \forall (u,v)\in E_\tau\ . \tag{7.28}$$

Adding them together gives

$$(2n_\tau-1)\|P_{\boldsymbol{\mu}_\tau}-P_{\boldsymbol{\nu}_\tau}\|_1 \geq \|\boldsymbol{\mu}_\tau-\boldsymbol{\nu}_\tau\|_1 \geq \|\boldsymbol{\mu}_\tau-\boldsymbol{\nu}_\tau\|_2\ . \tag{7.29}$$

According to Pinsker's inequality [27], we have

$$\begin{aligned} d_\phi(\boldsymbol{\mu}_\tau\|\boldsymbol{\nu}_\tau) = KL(P_{\boldsymbol{\mu}_\tau}, P_{\boldsymbol{\nu}_\tau}) &\geq \frac{1}{2}\|P_{\boldsymbol{\mu}_\tau}-P_{\boldsymbol{\nu}_\tau}\|_1^2 \\ &\geq \frac{1}{2(2n_\tau-1)^2}\|\boldsymbol{\mu}_\tau-\boldsymbol{\nu}_\tau\|_2^2\ . \end{aligned} \tag{7.30}$$

Multiplying α on both sides and letting $\alpha \geq \beta(2n_\tau-1)^2$ complete the proof. ■

To prove the convergence of the objective function, we define a residual term R_τ^{t+1} as

$$R_\tau^{t+1} = \rho_\tau\langle \boldsymbol{m}_\tau^{t+1}-\boldsymbol{\mu}_\tau^*, \boldsymbol{\theta}_\tau\rangle\ , \tag{7.31}$$

where $\boldsymbol{\mu}_\tau^*$ is the optimal solution for tree τ. We show that R_τ^{t+1} satisfies the following inequality:

Lemma 22 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. Assume $\alpha \geq \max_\tau\{\beta(2n_\tau-1)^2\}$. For any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have*

$$\begin{aligned} R_\tau^{t+1} \leq{} & \langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^*-\boldsymbol{m}_\tau^{t+1}\rangle + \alpha\big(d_\phi(\boldsymbol{\mu}_\tau^*\|\boldsymbol{m}_\tau^t)-d_\phi(\boldsymbol{\mu}_\tau^*\|\boldsymbol{m}_\tau^{t+1})\big) \\ &+\frac{\beta}{2}\big(\|\boldsymbol{\mu}_\tau^*-\boldsymbol{\mu}_\tau^t\|_2^2-\|\boldsymbol{\mu}_\tau^*-\boldsymbol{m}_\tau^t\|_2^2-\|\boldsymbol{m}_\tau^{t+1}-\boldsymbol{\mu}_\tau^t\|_2^2\big)\ , \end{aligned} \tag{7.32}$$

where R_τ^{t+1} is defined in (7.31).

Proof: Since $\boldsymbol{m}_\tau^{t+1}$ is the optimal solution for (7.21), for any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have the following inequality:

$$\langle \mathbf{y}_r^t + \alpha(\nabla\phi(\boldsymbol{m}_\tau^{t+1})-\nabla\phi(\boldsymbol{m}_\tau^t)), \boldsymbol{\mu}_\tau^*-\boldsymbol{m}_\tau^{t+1}\rangle \geq 0\ . \tag{7.33}$$

Substituting (7.16) into (7.33) and rearranging the terms, we have

$$R_\tau^{t+1} \leq \langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^*-\boldsymbol{m}_\tau^{t+1}\rangle + \beta\langle \boldsymbol{m}_\tau^t-\boldsymbol{\mu}_\tau^t, \boldsymbol{\mu}_\tau^*-\boldsymbol{m}_\tau^{t+1}\rangle$$

$$+ \alpha\langle\nabla\phi(\boldsymbol{m}_\tau^{t+1}) - \nabla\phi(\boldsymbol{m}_\tau^{t}), \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\rangle \ . \tag{7.34}$$

The second term in the RHS of (7.34) is equivalent to

$$2\langle\boldsymbol{m}_\tau^t - \boldsymbol{\mu}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\rangle = \|\boldsymbol{m}_\tau^t - \boldsymbol{m}_\tau^{t+1}\|_2^2$$
$$+\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^t\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^t\|_2^2 - \|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2. \tag{7.35}$$

The third term in the RHS of (7.34) can be rewritten as

$$\langle\nabla\phi(\boldsymbol{m}_\tau^{t+1}) - \nabla\phi(\boldsymbol{m}_\tau^{t}), \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\rangle$$
$$= d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^t) - d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^{t+1}) - d_\phi(\boldsymbol{m}_\tau^{t+1} || \boldsymbol{m}_\tau^t). \tag{7.36}$$

Substituting (7.35) and (7.36) into (7.34) and using Lemma 21 complete the proof. ■

We next show that the first term in the RHS of (7.32) satisfies the following result:

Lemma 23 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. For any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have*

$$\sum_{\tau=1}^{|\mathbb{T}|}\langle\boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\rangle \leq \frac{1}{2\beta}(\|\boldsymbol{\lambda}_\tau^t\|_2^2 - \|\boldsymbol{\lambda}_\tau^{t+1}\|_2^2)$$
$$+ \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^{t+1}\|_2^2\right) \ .$$

Proof: Let μ_i be the ith component of $\boldsymbol{\mu}$. We augment $\boldsymbol{\mu}_\tau, \boldsymbol{m}_\tau$ and $\boldsymbol{\lambda}_\tau$ in the following way: If $\boldsymbol{\mu}_i$ is not a component of $\boldsymbol{\mu}_\tau$, we set $\boldsymbol{\mu}_{\tau,i} = 0, \boldsymbol{m}_{\tau,i} = 0$ and $\boldsymbol{\lambda}_{\tau,i} = 0$; otherwise, they are the corresponding components from $\boldsymbol{\mu}_\tau, \boldsymbol{m}_\tau$ and $\boldsymbol{\lambda}_\tau$ respectively. We can then rewrite (7.22) in the following equivalent component-wise form:

$$\mu_i^{t+1} = \text{argmin}_{\mu_i}\sum_{\tau=1}^{|\mathbb{T}|}\left(\langle\lambda_{\tau,i}^t, m_{\tau,i}^{t+1} - \mu_{\tau,i}\rangle + \frac{\beta}{2}||m_{\tau,i}^{t+1} - \mu_{\tau,i}||_2^2\right) \ .$$

For any $\boldsymbol{\mu}_\tau^* \in L(\tau)$, we have the following optimality condition:

$$-\sum_{\tau=1}^{|\mathbb{T}|}\langle\lambda_{\tau,i}^t + \beta(m_{\tau,i}^{t+1} - \mu_{\tau,i}^{t+1}), \mu_{\tau,i}^* - \mu_{\tau,i}^{t+1}\rangle \geq 0 \ . \tag{7.37}$$

Combining all the components of $\boldsymbol{\mu}^{t+1}$, we can rewrite (7.37) in the following vector form:

$$-\sum_{\tau=1}^{|\mathbb{T}|}\langle\boldsymbol{\lambda}_\tau^t + \beta(\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}), \boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^{t+1}\rangle \geq 0 \ . \tag{7.38}$$

Rearranging the terms yields

$$
\begin{aligned}
&\sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\rangle \\
&\leq \sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1}\rangle - \sum_{\tau=1}^{|\mathbb{T}|}\beta\langle \boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}, \boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^{t+1}\rangle \\
&= \sum_{\tau=1}^{|\mathbb{T}|}\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1}\rangle + \frac{\beta}{2}\sum_{\tau=1}^{|\mathbb{T}|}\big(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\|_2^2 \\
&-\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^{t+1}\|_2^2 - \|\boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1}\|_2^2\big) \ . \qquad (7.39)
\end{aligned}
$$

Recall $\boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1} = \frac{1}{\beta}(\boldsymbol{\lambda}_\tau^t - \boldsymbol{\lambda}_\tau^{t+1})$ in (7.23), then

$$
\langle \boldsymbol{\lambda}_\tau^t, \boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1}\rangle - \frac{\beta}{2}\|\boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{m}_\tau^{t+1}\|_2^2 = \frac{1}{2\beta}(\|\boldsymbol{\lambda}_\tau^t\|_2^2 - \|\boldsymbol{\lambda}_\tau^{t+1}\|_2^2) \ . \qquad (7.40)
$$

Plugging (7.40) into (7.39) completes the proof. ■

We also need the following lemma which can be found in [65]. We omit the proof due to lack of space.

Lemma 24 *Let $\{\boldsymbol{m}_\tau, \boldsymbol{\mu}_\tau, \boldsymbol{\lambda}_\tau\}$ be the sequences generated by Bethe-ADMM. Then*

$$
\sum_{\tau=1}^{|\mathbb{T}|}\|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2 \geq \sum_{\tau=1}^{|\mathbb{T}|}\|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^{t+1}\|_2^2 + \|\boldsymbol{\mu}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2 \ .
$$

Theorem 27 *Assume the following hold: (1) $\boldsymbol{m}_\tau^0$ and $\boldsymbol{\mu}_\tau^0$ are uniform tree-structured distributions, $\forall \tau = 1, \ldots, |\mathbb{T}|$ (2) $\boldsymbol{\lambda}_\tau^0 = \mathbf{0}, \forall \tau = 1, \ldots, |\mathbb{T}|$; (3) $\max_\tau d_\phi(\boldsymbol{\mu}_\tau^*||\boldsymbol{m}_\tau^0) = D_{\boldsymbol{\mu}}$; (4) $\alpha \geq \max_\tau\{\beta(2n_\tau - 1)^2\}$ holds. Denote $\bar{\boldsymbol{m}}_\tau^T = \frac{1}{T}\sum_{t=0}^{T-1}\boldsymbol{m}_\tau^t$ and $\bar{\boldsymbol{\mu}}_\tau^T = \frac{1}{T}\sum_{t=0}^{T-1}\boldsymbol{\mu}_\tau^t$. For any T and the optimal solution $\boldsymbol{\mu}^*$, we have*

$$
\sum_{\tau=1}^{|\mathbb{T}|}\left(\rho_\tau\langle \bar{\boldsymbol{m}}_\tau^T - \boldsymbol{\mu}_\tau^*, \theta_\tau\rangle + \frac{\beta}{2}\|\bar{\boldsymbol{m}}_\tau^T - \bar{\boldsymbol{\mu}}_\tau^T\|_2^2\right) \leq \frac{D_{\boldsymbol{\mu}}\alpha|\mathbb{T}|}{T} \ .
$$

Proof: Summing (7.32) over τ from 1 to $|\mathbb{T}|$ and using Lemma 23, we have:

$$
\sum_{\tau=1}^{|\mathbb{T}|}\left(R_\tau^{t+1} + \frac{\beta}{2}\|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2\right)
$$

$$\leq \sum_{\tau=1}^{|\mathbb{T}|} \frac{1}{2\beta}(\|\boldsymbol{\lambda}_\tau^t\|_2^2 - \|\boldsymbol{\lambda}_\tau^{t+1}\|_2^2) + \frac{\beta}{2}(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^t\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^{t+1}\|_2^2)$$
$$+ \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^{t+1}\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^t\|_2^2\right)$$
$$+ \alpha\left(d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^t) - d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^{t+1})\right) . \tag{7.41}$$

Summing over the above from $t = 0$ to $T - 1$, we have

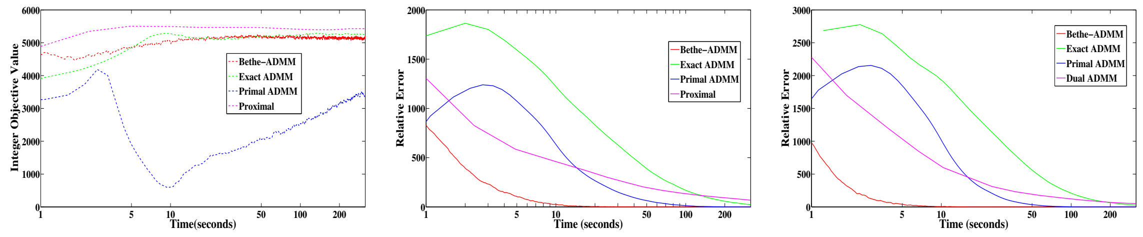
$$\sum_{t=0}^{T-1} \sum_{\tau=1}^{|\mathbb{T}|} \left(R_\tau^{t+1} + \frac{\beta}{2}\|\boldsymbol{m}_\tau^{t+1} - \boldsymbol{\mu}_\tau^t\|_2^2 \right)$$
$$\leq \sum_{\tau=1}^{|\mathbb{T}|} \frac{1}{2\beta}(\|\boldsymbol{\lambda}_\tau^0\|_2^2 - \|\boldsymbol{\lambda}_\tau^T\|_2^2) + \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^0\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{\mu}_\tau^T\|_2^2\right)$$
$$+ \frac{\beta}{2}\left(\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^T\|_2^2 - \|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^0\|_2^2\right)$$
$$+ \alpha\left(d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^0) - d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^T)\right)$$
$$\leq \sum_{\tau=1}^{|\mathbb{T}|} \frac{\beta}{2}\|\boldsymbol{\mu}_\tau^* - \boldsymbol{m}_\tau^T\|_2^2 + \alpha\left(d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^0) - d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^T)\right)$$
$$\leq \sum_{\tau=1}^{|\mathbb{T}|} \alpha d_\phi(\boldsymbol{\mu}_\tau^* || \boldsymbol{m}_\tau^0) , \tag{7.42}$$

where we use Lemma 21 to derive (7.42). Applying Lemma 24 and Jensen's inequality yield the desired bound. ■

Theorem 1 establishes the $O(1/T)$ convergence rate for the Bethe-ADMM in ergodic sense. As $T \to \infty$, the objective value $\sum_{\tau=1}^{|\mathbb{T}|} \rho_\tau \langle \bar{\boldsymbol{m}}_\tau^T, \theta_\tau \rangle$ converges to the optimal value and the equality constraints are also satisfied.

7.3.4 Extension to MRFs with General Factors

Although we present Bethe-ADMM in the context of pairwise MRFs, it can be easily generalized to handle MRFs with general factors. For a general MRF, we can view the dependency graph as a factor graph [107], a bipartite graph $G = (V \cup F, E)$, where V and F are disjoint set of variable nodes and factor nodes and E is a set of edges, each connecting a variable node and a factor node. The distribution $P(\mathbf{x})$ takes the form: $P(\mathbf{x}) \propto \exp\left\{\sum_{u \in V} f_u(x_u) + \sum_{\alpha \in F} f_\alpha(\mathbf{x}_\alpha)\right\}$.



(a) Rounded solution with $a = 0.5$. (b) Relative error with $a = 0.5$. (c) Relative error with $a = 1$.

Figure 7.1: Results of Bethe-ADMM, Exact ADMM, Primal ADMM and proximal algorithms on two simulation datasets. Figure 7.1(a) plots the value of the decoded integer solution as a function of runtime (seconds). Figure 7.1(b) and 7.1(c) plot the relative error with respect to the optimal LP objective as a function of runtime (seconds). For Bethe-ADMM, we set $\alpha = \beta = 0.05$. For Exact ADMM, we set $\beta = 0.05$. For Primal ADMM, we set $\beta = 0.5$. Bethe-ADMM converges faster than other primal based algorithms.

The relaxed LP for general MRFs can be constructed in a similar fashion with that for pairwise MRFs.

We can then decompose the relaxed LP to subproblems defined on factor trees and impose equality constraints to enforce consistency on the shared variables among the subproblems. Each subproblem can be solved efficiently using the sum-product algorithm for factor trees and the Bethe-ADMM algorithm for general MRFs bears similar structure with that for pairwise MRFs.

7.4 Experimental Results

We compare the Bethe-ADMM algorithm with several other state-of-the-art MAP inference algorithms. We show the comparison results with primal based MAP inference algorithms in Section 7.4.1 and dual based MAP inference algorithm in Section 7.4.2 respectively. We also show in Section 7.4.3 how tree decomposition benefits the performance of Bethe-ADMM. We run experiments in Section 7.4.1-7.4.3 using sequential updates. To illustrate the scalability of our algorithm, we run parallel Bethe-ADMM on a multicore machine and show the linear speedup in Section 7.4.4.

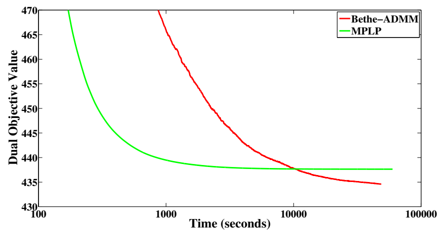


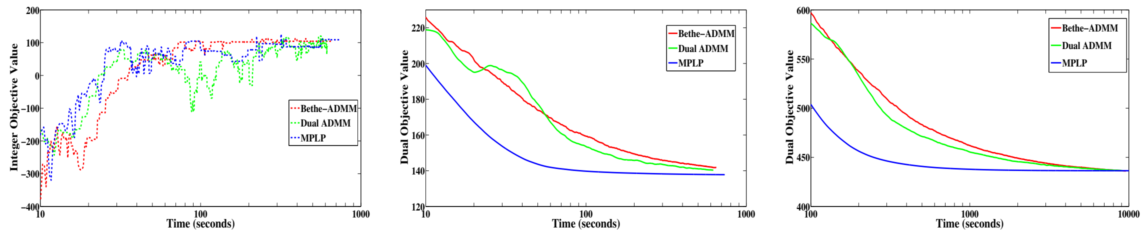
Figure 7.2: Both Bethe-ADMM and MPLP are run for sufficiently long, i.e., 50000 iterations. The dual objective value is plotted as a function of runtime (seconds). The MPLP algorithm gets stuck and does not reach the global optimum.

7.4.1 Comparison with Primal based Algorithms

We compare the Bethe-ADMM algorithm with the proximal algorithm [158], Exact ADMM algorithm and Primal ADMM algorithm [139]. For the proximal algorithm, we choose the Bregman divergence as the sum of KL-divergences across all node and edge distributions. Following the methodology in [158], we terminate the inner loop if the maximum constraint violation of $L(G)$ is less than 10^{-3} and set $w^t = t$. Similarly, in applying the Exact ADMM algorithm, we terminate the loop for solving $\mathbf{M}_\tau$ if the maximum constraint violation of $L(\tau)$ is less than 10^{-3}. For the Exact ADMM and Bethe-ADMM algorithm, we use 'edge decomposition': each τ is simply an edge of the graph and $|\mathbb{T}| = |E|$. To obtain the integer solution, we use node-based rounding: $x_u^* = \text{argmax}_{x_u} \mu_u(x_u)$.

We show experimental results on two synthetic datasets. The underlying graph of each dataset is a three dimensional $m \times n \times t$ grid. We generate the potentials as follows: We set the nodewise potentials as random numbers from $[-a, a]$, where $a > 0$. We set the edgewise potentials according to the Potts model, i.e., $\theta_{uv}(x_u, x_v) = b_{uv}$ if $x_u = x_v$ and 0 otherwise. We choose b_{uv} randomly from $[-1, 1]$. The edgewise potentials penalize disagreement if $b_{uv} > 0$ and penalize agreement if $b_{uv} < 0$. We generate datasets using $m = 20, n = 20, t = 16, k = 6$ with varying a.

Figure 7.1(a) shows the plots of (7.1) on one synthetic dataset and we find that the algorithms have similar performances on other simulation datasets. We observe that all algorithms converge to the optimal value $\langle \boldsymbol{\mu}^*, \boldsymbol{f} \rangle$ of (7.3) and we plot the relative error with respect to the optimal value $|\langle \boldsymbol{\mu}^* - \boldsymbol{\mu}_t, \boldsymbol{f} \rangle|$ on the two datasets in Figure 7.1(b) and 7.1(c).



(a) Rounded integer solution on 1jo8. (b) Dual value on 1jo8. (c) Dual value on 1or7.

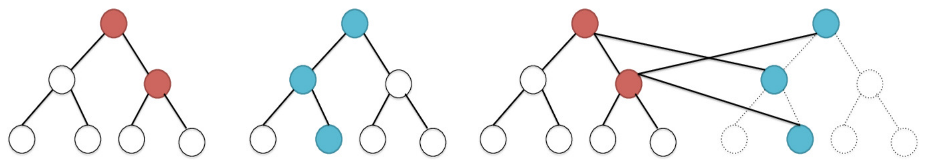
Figure 7.3: Results of Bethe-ADMM, MPLP and Dual ADMM algorithms on two protein design datasets. Figure 7.3(a) plots the the value of the decoded integer solution as a function of runtime (seconds). Figure 7.3(b) and 7.3(c) plot the dual value as a function of runtime (seconds). For Dual ADMM, we set $\beta = 0.05$. For Bethe-ADMM, we set $\alpha = \beta = 0.1$. Bethe-ADMM and Dual ADMM have similar performance in terms of convergence. All three methods have comparable performances for the decoded integer solution.

Overall, the Bethe-ADMM algorithm converges faster than other primal algorithms. We observe that the proximal algorithm and Exact ADMM algorithm are the slowest, due to the sequential projection step. In terms of the decoded integer solution, the Bethe-ADMM, Exact ADMM and proximal algorithm have similar performances. We also note that a higher objective function value does not necessarily lead to a better decoded integer solution.

7.4.2 Comparison with Dual based Algorithms

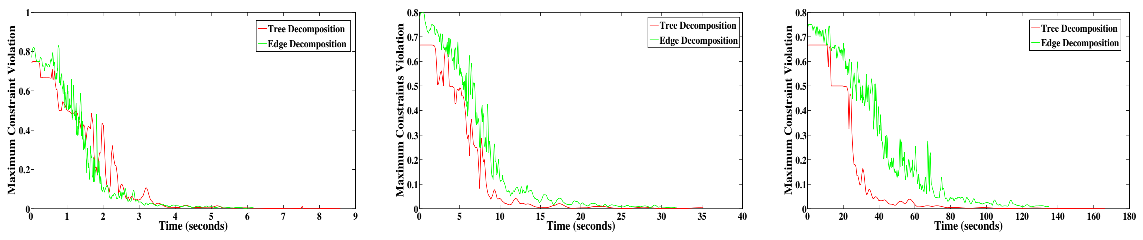
In this section, we compare the Bethe-ADMM algorithm with the MPLP algorithm [68] and the Dual ADMM algorithm [139]. We conduct experiments on protein design problems [215]. In these problems, we are given a 3D structure and the goal is to find a sequence of amino-acids that is the most stable for that structure. The problems are modeled by nodewise and pairwise factors and can be posed as finding a MAP assignment for the given model. This is a demanding setting in which each problem may have hundreds of variables with 100 possible states on average.

We run the algorithms on two problems with different sizes [215], i.e., 1jo8 (58 nodes and 981 edges) and 1or7 (180 nodes and 3005 edges). For the MPLP and Dual ADMM algorithm, we plot the value of the integer programming problem (7.1) and its dual.. For Bethe-ADMM algorithm, we plot the value of dual LP of (7.3) and the integer programming problem (7.1).



(a) (b)

Figure 7.4: A simulation dataset with $m = 2$, $s = 7$ and $n = 3$. In 7.4(a), the red nodes (S_{12}) are sampled from tree 1 and the blue nodes (D_{12}) are sampled from tree 2. In 7.4(b) , sampled nodes are connected by cross-tree edges (E_{12}). Tree 1 with nodes in D_{12} and edges in E_{12} still form a tree, denoted by solid lines. This augmented tree is a tree-structured subgraph for Bethe-ADMM.



(a) $s = 1023$. (b) $s = 4095$. (c) $s = 16383$.

Figure 7.5: Results of Bethe-ADMM algorithms based on tree and edge decomposition on three simulation datasets with $m = 10, n = 20$. The maximum constraint violation in $L(G)$ is plotted as a function of runtime (seconds). For both algorithms, we set $\alpha = \beta = 0.05$. The tree based Bethe-ADMM algorithm has better performance than that of the edge based Bethe-ADMM when the tree structure is more dominant in G.

Note that although Bethe-ADMM and Dual ADMM have different duals, their optimal values are the same. We run the Bethe-ADMM based on edge decomposition. Figure 7.3 shows the result.

We observe that the MPLP algorithm usually converges faster, but since it is a coordinate ascent algorithm, it can stop prematurely and yield suboptimal solutions. Figure 7.2 shows that on the 1fpo dataset, the MPLP algorithm converges to a suboptimal solution. We note that the convergence time of the Bethe-ADM and Dual ADM are similar. The three algorithms have similar performance in terms of the decoded integer solution.

7.4.3 Edge based vs Tree based

In the previous experiments, we use ‘edge decomposition’ for the Bethe-ADMM algorithm. Since our algorithm can work for any tree-structured graph decomposition, we want to empirically study how the decomposition affects the performance of the Bethe-ADMM algorithm. In the following experiments, we show that if we can utilize the graph structure when decomposing the graph, the Bethe-ADMM algorithm will have better performance compared to simply using ‘edge decomposition’, which does not take the graph structure into account.

We conduct experiments on synthetic datasets. We generate MRFs whose dependency graphs consist of several tree-structured graphs and cross-tree edges to introduce cycles. To be more specific, we first generate m binary tree structured MRFs each with s nodes. Then for each ordered pair of tree-structured MRFs $(i, j), 1 \leq i, j \leq m, i \neq j$, we uniformly sample n nodes from MRF i with replacement and uniformly sample n $(n \leq s)$ nodes from MRF j without replacement, resulting in two node sets S_{ij} and D_{ij}. We then connect the nodes in S_{ij} and D_{ij}, denoting them as E_{ij}. We repeat this process for every pair of trees. By construction, the graph consisting of tree i, nodes in D_{ij} and edges in E_{ij}, $\forall j \neq i$ is still a tree. We will use these m augmented trees as the tree-structured subgraphs for the Bethe-ADMM algorithm. Figure 7.4 illustrates the graph generation and tree decomposition process. A simple calculation shows that for this particular tree decomposition, $O(m^2nk)$ equality constraints are maintained, while for edge decomposition, $O(msk + m^2nk)$ are maintained. When the graph has dominant tree structure, tree decomposition leads to much less number of equality constraints.

For the experiments, we run the Bethe-ADMM algorithm based on tree and edge decomposition with different values of s, keeping m and n fixed. It is easy to see that the tree structure becomes more dominant when s becomes larger. Since we observe that both algorithms first converge to the optimal value of (7.3) and then the equality constraints are gradually satisfied, we evaluate the performance by computing the maximum constraint violation of $L(G)$ at each iteration for both algorithms. The faster the constraints are satisfied, the better the algorithm is. The results are shown in Figure 7.5. When the tree structure is not obvious, the two algorithms have similar performances. As we increase s and the tree structure becomes more dominant, the difference between the two algorithms is more pronounced. We attribute the superior performance to the fact that for the tree decomposition case, much fewer number of equality constraints are imposed and each subproblem on tree can be solved efficiently using the sum-product algorithm.

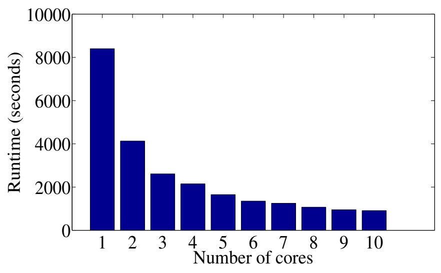


Figure 7.6: The Open MPI implementation of Bethe-ADMM has almost linear speedup on the CRU dataset with more than 7 million nodes.

7.4.4 Scalability Experiments on Multicores

The dataset used in this section is the Climate Research Unit (CRU) precipitation dataset [140], which has monthly precipitation from the years 1901-2006. The dataset is of high gridded spatial resolution (360×720, i.e., 0.5 degree latitude $\times$ 0.5 degree longitude) and includes the precipitation over land.

Our goal is to detect major droughts based on precipitation. We formulate the problem as the one of estimating the most likely configuration of a binary MRF, where each node represents a location. The underlying graph is a three dimensional grid ($360 \times 720 \times 106$) with 7,146,520 nodes and each node can be in two possible states: dry and normal. We run the Bethe-ADMM algorithm on the CRU dataset and detect droughts based on the integer solution after node-based rounding. For the details of the this experiment, we refer to readers to [63]. Our algorithm successfully detects nearly all the major droughts of the last century. We also examine how the Bethe-ADMM algorithm scales on the CRU dataset with more than 7 million variables. We run the Open MPI code with different number of cores and the result in Figure 7.6 shows that we obtain almost linear speedup with the number of cores.

Chapter 8

Large Scale Sparse Precision Estimation

8.1 Introduction

Consider a p-dimensional probability distribution with true covariance matrix $\Sigma_0 \in \mathcal{S}^p_{++}$ and true precision (or inverse covariance) matrix $\Omega_0 = \Sigma_0^{-1} \in \mathcal{S}^p_{++}$. Let $[R_1 \ \cdots \ R_n] \in \Re^{p\times n}$ be n independent and identically distributed random samples drawn from this p-dimensional distribution. The centered normalized sample matrix $\mathbf{A} = [\mathbf{a}_1 \ \cdots \mathbf{a}_n] \in \Re^{p\times n}$ can be obtained as $\mathbf{a}_i = \frac{1}{\sqrt{n}}(R_i - \bar{R})$, where $\bar{R} = \frac{1}{n}\sum_i R_i$, so that the sample covariance matrix can be computed as $\mathbf{C} = \mathbf{A}\mathbf{A}^T$. In recent years, considerable effort has been invested in obtaining an accurate estimate of the precision matrix $\hat{\Omega}$ based on the sample covariance matrix $\mathbf{C}$ in the 'low sample, high dimensions' setting, i.e., $n \ll p$, especially when the true precision Ω_0 is assumed to be sparse [217]. Suitable estimators and corresponding statistical convergence rates have been established for a variety of settings, including distributions with sub-Gaussian tails, polynomial tails [159, 23, 120]. Recent advances have also established parameter-free methods which achieve minimax rates of convergence [22, 120].

Spurred by these advances in the statistical theory of precision matrix estimation, there has been considerable recent work on developing computationally efficient *optimization methods* for solving the corresponding statistical estimation problems: see [7, 59, 89, 134, 88], and references therein. While these methods are able to efficiently solve problems up to a few thousand variables, ultra-large-scale problems with millions of variables remain a challenge.

Note further that in precision matrix estimation, the number of parameters scales quadratically with the number of variables; so that with a million dimensions $p = 10^6$, the total number of parameters to be estimated is a trillion, $p^2 = 10^{12}$. The focus of this chapter is on designing an efficient distributed algorithm for precision matrix estimation under such ultra-large-scale dimensional settings.

We focus on the CLIME statistical estimator [23], which solves the following linear program (LP):

$$\min \|\hat{\Omega}\|_1 \quad \text{s.t.} \quad \|\mathbf{C}\hat{\Omega} - \mathbf{I}\|_\infty \leq \lambda \,, \tag{8.1}$$

where $\lambda > 0$ is a tuning parameter. The CLIME estimator not only has strong statistical guarantees [23], but also comes with inherent computational advantages. First, the LP in (9.5) does not explicitly enforce positive definiteness of $\hat{\Omega}$, which can be a challenge to handle efficiently in high-dimensions. Secondly, it can be seen that (9.5) can be decomposed into p independent LPs, one for each column of $\hat{\Omega}$. This separable structure has motivated solvers for (9.5) which solve the LP column-by-column using interior point methods [23, 217] or the alternating direction method of multipliers (ADMM) [114]. However, these solvers do not scale well to ultra-high-dimensional problems: they are not designed to run on hundreds to thousands of cores, and in particular require the entire sample covariance matrix $\mathbf{C}$ to be loaded into the memory of a single machine, which is impractical even for moderate sized problems.

In this chapter, we present an efficient CLIME-ADMM variant along with a scalable distributed framework for the computations [19, 200]. The proposed CLIME-ADMM algorithm can scale up to millions of dimensions, and can use up to thousands of cores in a shared-memory or distributed-memory architecture. The scalability of our method relies on the following key innovations. First, we propose an inexact ADMM [214, 79] algorithm targeted to CLIME, where each step is either elementwise parallel or involves suitable matrix multiplications. We show that the rates of convergence of the objective to the optimum as well as residuals of constraint violation are both $O(1/T)$. Second, we solve (9.5) in column-blocks of the precision matrix at a time, rather than one column at a time. Since (9.5) already decomposes columnwise, solving multiple columns together in blocks might not seem worthwhile. However, as we show our CLIME-ADMM working with column-blocks uses matrix-matrix multiplications which, building on existing literature [16, 36, 73] and the underlying low rank and sparse structure inherent in the precision matrix estimation problem, can be made substantially more efficient than repeated matrix-vector multiplications. Moreover, matrix multiplication can be further simplified as block-by-block operations, which allows choosing optimal block sizes to minimize cache misses, leading to high scalability and performance [109, 36, 16]. Lastly, since the core computations can be parallelized, CLIME-ADMM scales almost linearly with the number of cores. We experiment with shared-memory and distributed-memory architectures to illustrate this point. Empirically, CLIME-ADMM is shown to be much faster than existing methods for precision estimation, and scales well to high-dimensional problems, e.g., we estimate a precision matrix of one million dimension and one trillion parameters in 11 hours by running the algorithm on 400 cores.

Our framework can be positioned as a part of the recent surge of effort in scaling up machine learning algorithms [226, 150, 42, 43, 123, 19, 154, 64] to “Big Data”. Scaling up machine learning algorithms through parallelization and distribution has been heavily explored on various architectures, including shared-memory architectures [150], distributed memory architectures [154, 42, 64] and GPUs [157]. Since MapReduce [43] is not efficient for optimization algorithms, [42] proposed a parameter server that can be used to parallelize gradient descent algorithms for unconstrained optimization problems. However, this framework is ill-suited for the constrained optimization problems we consider here, because gradient descent methods require the projection at each iteration which involves all variables and thus ruins the parallelism. In other recent related work based on ADMM, [154] introduce graph projection block splitting (GPBS) to split data into blocks so that examples and features can be distributed among multiple cores. Our framework uses a more general blocking scheme (block cyclic distribution), which provides more options in choosing the optimal block size to improve the efficiency in the use of memory hierarchies and minimize cache misses [109, 16, 36]. ADMM has also been used to solve constrained optimization in a distributed framework [64] for graphical model inference, but they consider local constraints, in contrast to the global constraints in our framework.

Notation: A matrix is denoted by a bold face upper case letter, e.g., $\mathbf{A}$. An element of a matrix is denoted by a upper case letter with row index i and column index j, e.g., A_{ij} is the ij-th element of $\mathbf{A}$. A block of matrix is denoted by a bold face lower case letter indexed by ij, e.g., $\mathbf{A}_{ij}$. $\vec{\mathbf{A}}_{ij}$ represents a collection of blocks of matrix $\mathbf{A}$ on the ij-th core (see block cyclic distribution in Section 4). $\mathbf{A}'$ refers the transpose of $\mathbf{A}$. Matrix norms used are all elementwise norms, e.g., $\|\mathbf{A}\|_1 = \sum_{i=1}^{p} \sum_{j=1}^{n} |A_{ij}|$, $\|\mathbf{A}\|_2^2 = \sum_{i=1}^{p} \sum_{j=1}^{n} A_{ij}^2$, $\|\mathbf{A}\|_\infty =$

Algorithm 6 Column Block ADMM for CLIME

```
1:  Input: $\mathbf{C}, \lambda, \rho, \eta$
2:  Output: $\mathbf{X}$
3:  Initialization: $\mathbf{X}^0, \mathbf{Z}^0, \mathbf{Y}^0, \mathbf{V}^0, \hat{\mathbf{V}}^0 = 0$
4:  for $t = 0$ to $T-1$ do
5:      X-update: $\mathbf{X}^{t+1} = \textit{soft}(\mathbf{X}^t - \mathbf{V}^t, \frac{1}{\eta})$, where $\textit{soft}(\mathbf{X}, \gamma) = \begin{cases} X_{ij} - \gamma\,, & \text{if } X_{ij} > \gamma\,, \\ X_{ij} + \gamma\,, & \text{if } X_{ij} < -\gamma\,, \\ 0\,, & \text{otherwise} \end{cases}$
6:      Mat-Mul: $\begin{cases} \text{sparse}: & \mathbf{U}^{t+1} = \mathbf{C}\mathbf{X}^{t+1} \\ \text{low rank}: & \mathbf{U}^{t+1} = \mathbf{A}(\mathbf{A}'\mathbf{X}^{t+1}) \end{cases}$
7:      Z-update: $\mathbf{Z}^{t+1} = \textit{box}(\mathbf{U}^{t+1} + \mathbf{Y}^t, \lambda)$, where $\textit{box}(\mathbf{X}, \mathbf{E}, \lambda) = \begin{cases} E_{ij} + \lambda, & \text{if } X_{ij} - E_{ij} > \lambda, \\ X_{ij}, & \text{if } |X_{ij} - E_{ij}| \le \lambda, \\ E_{ij} - \lambda, & \text{if } X_{ij} - E_{ij} < -\lambda, \end{cases}$
8:      Y-update: $\mathbf{Y}^{t+1} = \mathbf{Y}^t + \mathbf{U}^{t+1} - \mathbf{Z}^{t+1}$
9:      Mat-Mul: $\begin{cases} \text{sparse}: & \hat{\mathbf{V}}^{t+1} = \mathbf{C}\mathbf{Y}^{t+1} \\ \text{low rank}: & \hat{\mathbf{V}}^{t+1} = \mathbf{A}(\mathbf{A}'\mathbf{Y}^{t+1}) \end{cases}$
10:     V-update: $\mathbf{V}^{t+1} = \frac{\rho}{\eta}(2\hat{\mathbf{V}}^{t+1} - \hat{\mathbf{V}}^t)$
11: end for
```

$\max_{1 \le i \le p, 1 \le j \le n} |A_{ij}|$. The matrix inner product is defined in elementwise, e.g., $\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i=1}^{p} \sum_{j=1}^{n} A_{ij} B_{ij}$. $\mathbf{X} \in \Re^{p \times k}$ denotes $k (1 \le k \le p)$ columns of the precision matrix $\hat{\Omega}$, and $\mathbf{E} \in \Re^{p \times k}$ denotes the same k columns of the identity matrix $\mathbf{I} \in \Re^{p \times p}$. Let $\lambda_{\max}(\mathbf{C})$ be the largest eigenvalue of covariance matrix $\mathbf{C}$.

8.2 Column Block ADMM for CLIME

In this section, we propose an algorithm to estimate the precision matrix in terms of column blocks instead of column-by-column. Assuming a column block contains $k (1 \le k \le p)$ columns, the sparse precision matrix estimation amounts to solving $\lceil p/k \rceil$ independent linear programs. Denoting $\mathbf{X} \in \Re^{p \times k}$ be k columns of $\hat{\Omega}$, (9.5) can be written as

$$\min \|\mathbf{X}\|_1 \quad \text{s.t.} \quad \|\mathbf{C}\mathbf{X} - \mathbf{E}\|_\infty \le \lambda\,, \tag{8.2}$$

which can be rewritten in the following equality-constrained form:

$$\min \|\mathbf{X}\|_1 \quad \text{s.t.} \quad \|\mathbf{Z} - \mathbf{E}\|_\infty \le \lambda, \mathbf{C}\mathbf{X} = \mathbf{Z}\,. \tag{8.3}$$

Through the splitting variable $\mathbf{Z} \in \Re^{p \times k}$, the infinity norm constraint becomes a box constraint and is separated from the ℓ_1 norm objective. We use ADMM to solve (8.3). The augmented

Lagrangian of (8.3) is

$$L_\rho = \|\mathbf{X}\|_1 + \rho\langle \mathbf{Y}, \mathbf{CX} - \mathbf{Z}\rangle + \frac{\rho}{2}\|\mathbf{CX} - \mathbf{Z}\|_2^2 , \tag{8.4}$$

where $\mathbf{Y} \in \Re^{p\times k}$ is a scaled dual variable and $\rho > 0$. ADMM yields the following iterates [19]:

$$\mathbf{X}^{t+1} = \text{argmin}_{\mathbf{X}} \|\mathbf{X}\|_1 + \frac{\rho}{2}\|\mathbf{CX} - \mathbf{Z}^t + \mathbf{Y}^t\|_2^2 , \tag{8.5}$$

$$\mathbf{Z}^{t+1} = \underset{\|\mathbf{Z}-\mathbf{E}\|_\infty \le \lambda}{\text{argmin}} \frac{\rho}{2}\|\mathbf{CX}^{t+1} - \mathbf{Z} + \mathbf{Y}^t\|_2^2 , \tag{8.6}$$

$$\mathbf{Y}^{t+1} = \mathbf{Y}^t + \mathbf{CX}^{t+1} - \mathbf{Z}^{t+1} . \tag{8.7}$$

As a Lasso problem, (8.5) can be solved using exisiting Lasso algorithms, but that will lead to a double-loop algorithm. (8.5) does not have a closed-form solution since C in the quadratic penalty term makes $\mathbf{X}$ coupled. We decouple $\mathbf{X}$ by linearizing the quadratic penalty term and adding a proximal term as follows:

$$\mathbf{X}^{t+1} = \text{argmin}_{\mathbf{X}} \|\mathbf{X}\|_1 + \eta\langle \mathbf{V}^t, \mathbf{X}\rangle + \frac{\eta}{2}\|\mathbf{X} - \mathbf{X}^t\|_2^2 , \tag{8.8}$$

where $\mathbf{V}^t = \frac{\rho}{\eta}\mathbf{C}(\mathbf{Y}^t + \mathbf{CX}^t - \mathbf{Z}^t)$ and $\eta > 0$. (8.8) is usually called an inexact ADMM update. Using (8.7), $\mathbf{V}^t = \frac{\rho}{\eta}\mathbf{C}(2\mathbf{Y}^t - \mathbf{Y}^{t-1})$. Let $\hat{\mathbf{V}}^t = \mathbf{CY}^t$, we have $\mathbf{V}^t = \frac{\rho}{\eta}(2\hat{\mathbf{V}}^t - \hat{\mathbf{V}}^{t-1})$. (8.8) has the following closed-form solution:

$$\mathbf{X}^{t+1} = \textit{soft}(\mathbf{X}^t - \mathbf{V}^t, \frac{1}{\eta}) , \tag{8.9}$$

where *soft* denotes the soft-thresholding and is defined in Step 5 of Algorithm 7.

Let $\mathbf{U}^{t+1} = \mathbf{CX}^{t+1}$. (8.6) is a box constrained quadratic programming which has the following closed-form solution:

$$\mathbf{Z}^{t+1} = \textit{box}(\mathbf{U}^{t+1} + \mathbf{Y}^t, \mathbf{E}, \lambda) , \tag{8.10}$$

where *box* denotes the projection onto the infinity norm constraint $\|\mathbf{Z}-\mathbf{E}\|_\infty \le \lambda$ and is defined in Step 7 of Algorithm 7. In particular, if $\|\mathbf{U}^{t+1} + \mathbf{Y}^t - \mathbf{E}\|_\infty \le \lambda$, $\mathbf{Z}^{t+1} = \mathbf{U}^{t+1} + \mathbf{Y}^t$ and thus $\mathbf{Y}^{t+1} = \mathbf{Y}^t + \mathbf{U}^{t+1} - \mathbf{Z}^{t+1} = \mathbf{0}$.

The ADMM algorithm for CLIME is summarized in Algorithm 7. In Algorithm 7, while step 5, 7, 8 and 10 amount to elementwise operations which cost $O(pk)$ operations, steps 6 and 9 involve matrix multiplication which is the most computationally intensive part and costs

$O(p^2k)$ operations. The memory requirement includes $O(pn)$ for $\mathbf{A}$ and $O(pk)$ for the other six variables.

As the following results show, Algorithm 1 has a $O(1/T)$ convergence rate for both the objective function and the residuals of optimality conditions. The proof technique is similar to [200]. [79] shows a similar result as Theorem 29 but uses a different proof technique. For proofs, please see Appendix A in the supplement.

Theorem 28 *Let* $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ *be generated by Algorithm 7 and* $\bar{\mathbf{X}}^T = \frac{1}{T}\sum_{t=1}^{T}\mathbf{X}^t$*. Assume* $\mathbf{X}^0 = \mathbf{Z}^0 = \mathbf{Y}^0 = \mathbf{0}$ *and* $\eta \geq \rho\lambda_{\max}^2(\mathbf{C})$*. For any* $\mathbf{CX} = \mathbf{Z}$*, we have*

$$\|\bar{\mathbf{X}}^T\|_1 - \|\mathbf{X}\|_1 \leq \frac{\eta\|\mathbf{X}\|_2^2}{2T} \, . \tag{8.11}$$

Theorem 29 *Let* $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ *be generated by Algorithm 7 and* $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ *be a KKT point for the Lagrangian of (8.3). Assume* $\mathbf{X}^0 = \mathbf{Z}^0 = \mathbf{Y}^0 = \mathbf{0}$ *and* $\eta \geq \rho\lambda_{\max}^2(\mathbf{C})$*. We have*

$$\|\mathbf{C}\mathbf{X}^T - \mathbf{Z}^T\|_2^2 + \|\mathbf{Z}^T - \mathbf{Z}^{T-1}\|_2^2 + \|\mathbf{X}^T - \mathbf{X}^{T-1}\|_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2}^2 \leq \frac{\|\mathbf{Y}^*\|_2^2 + \frac{\eta}{\rho}\|\mathbf{X}^*\|_2^2}{T} \, . \tag{8.12}$$

8.3 Leveraging Sparse, Low-Rank Structure

In this section, we consider a few possible directions that can further leverage the underlying structure of the problem; specifically sparse and low-rank structure.

8.3.1 Sparse Structure

As we detail here, there could be sparsity in the intermediate iterates, or the sample covariance matrix itself (or a perturbed version thereof); which can be exploited to make our CLIME-ADMM variant more efficient.

Iterate Sparsity: As the iterations progress, the soft-thresholding operation will yield a sparse $\mathbf{X}^{t+1}$, which can help speed up step 6: $\mathbf{U}^{t+1} = \mathbf{C}X^{t+1}$, via sparse matrix multiplication. Further, the box-thresholding operation will yield a sparse $\mathbf{Y}^{t+1}$. In the ideal case, if $\|\mathbf{U}^{t+1} + \mathbf{Y}^t - \mathbf{E}\|_\infty \leq \lambda$ in step 7, then $\mathbf{Z}^{t+1} = \mathbf{U}^{t+1} + \mathbf{Y}^t$. Thus, $\hat{\mathbf{Y}}^{t+1} = \mathbf{Y}^t + \mathbf{U}^{t+1} - \mathbf{Z}^{t+1} = \mathbf{0}$. More generally, $\mathbf{Y}^{t+1}$ will become sparse as the iterations proceed, which can help speed up step 9: $\hat{\mathbf{V}}^{t+1} = \mathbf{C}\mathbf{Y}^{t+1}$.

Sample Covariance Sparsity: We show that one can "perturb" the sample covariance to obtain a sparse and coarsened matrix, solve CLIME with this pertubed matrix, and yet have strong statistical guarantees. The statistical guarantees for CLIME [23], including convergence in spectral, matrix L_1, and Frobenius norms, only require from the sample covariance matrix $\mathbf{C}$ a deviation bound of the form $\|\mathbf{C} - \Sigma_0\|_\infty \leq c\sqrt{\log p/n}$, for some constant c. Accordingly, if we perturb the matrix $\mathbf{C}$ with a perturbation matrix Δ so that the perturbed matrix $(\mathbf{C} + \Delta)$ continues to satisfy the deviation bound, the statistical guarantees for CLIME would hold even if we used the perturbed matrix $(\mathbf{C}+\Delta)$. The following theorem (for details, please see Appendix B in the supplement) illustrates some perturbations Δ that satisfy this property:

Theorem 30 *Let the original random variables R_i be sub-Gaussian, with sample covariance $\mathbf{C}$. Let Δ be a random perturbation matrix, where Δ_{ij} are independent sub-exponential random variables. Then, for positive constants c_1, c_2, c_3, $P(\|\mathbf{C} + \Delta - \Sigma_0\|_\infty \geq c_1\sqrt{\frac{\log p}{n}}) \leq c_2 p^{-c_3}$.*

As a special case, one can thus perturb elements of C_{ij} with suitable constants Δ_{ij} with $|\Delta_{ij}| \leq c\sqrt{\log p/n}$, so that the perturbed matrix is sparse, i.e., if $|C_{ij}| \leq c\sqrt{\log p/n}$, then it can be safely truncated to 0. Thus, in practice, even if sample covariance matrix is only close to a sparse matrix [134, 88], or if it is close to being block diagonal [134, 88], the complexity of matrix multiplication in steps 6 and 9 can be significantly reduced via the above perturbations.

8.3.2 Low Rank Structure

Although one can use sparse structures of matrices participating in the matrix multiplication to accelerate the algorithm, the implementation requires substantial work since dynamic sparsity of $\mathbf{X}$ and $\mathbf{Y}$ is unknown upfront and static sparsity of the sample covariance matrix may not exist. Since the method will operate in a low-sample setting, we can alternatively use the *low rank* of the sample covariance matrix to reduce the complexity of matrix multiplication. Since $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ and $p \gg n$, $\mathbf{CX} = \mathbf{A}(\mathbf{A}^T\mathbf{X})$, and thus the computational complexity of matrix multiplication reduces from $O(p^2k)$ to $O(npk)$, which can achieve significant speedup for small n. We use such low-rank multiplications for the experiments in Section 8.5.

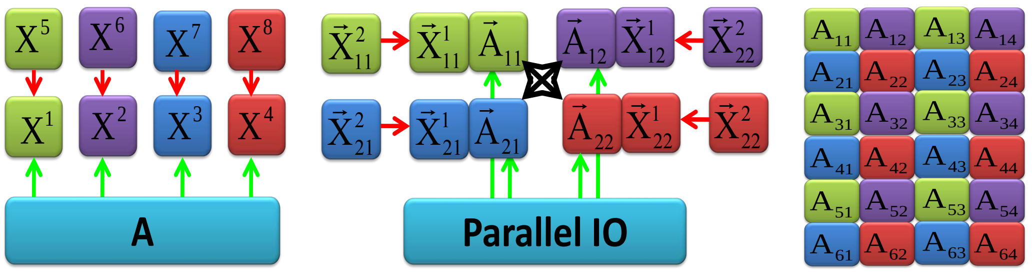
8.4 Scalable Parallel Computation Framework

In this section, we elaborate on scalable frameworks for CLIME-ADMM in both shared-memory and distributed-memory achitectures.

In a shared-memory architecture (e.g., a single machine), data $\mathbf{A}$ is loaded to the memory and shared by q cores, as shown in Figure 8.1(a). Assume the $p \times p$ precision matrix $\hat{\Omega}$ is evenly divided into $l = p/k$ $(\geq q)$ column blocks, e.g., $\mathbf{X}^1, \cdots, \mathbf{X}^q, \cdots, \mathbf{X}^l$, and thus each column block contains k columns. The column blocks are assigned to q cores cyclically, which means the j-th column block is assigned to the $mod(j, q)$-th core. The q cores can solve q column blocks in parallel without communication and synchronization, which can be simply implemented via multithreading. Meanwhile, another q column blocks are waiting in their respective queues. Figure 8.1(a) gives an example of how to solve 8 column blocks on 4 cores in a shared-memory environment. While the 4 cores are solving the first 4 column blocks, the next 4 column blocks are waiting in queues (red arrows).

Although the shared-memory framework is free from communication and synchronization, the limited resources prevent it from scaling up to datasets with millions of dimensions, which can not be loaded to the memory of a single machine or solved by tens of cores in a reasonble time. As more memory and computing power are needed for high dimensional datasets, we implement a framework for CLIME-ADMM in a distributed-memory architecture, which automatically distributes data among machines, parallelizes computation, and manages communication and synchronization among machines, as shown in Figure 8.1(b). Assume q processes are formed as a $r \times c$ process grid and the $p \times p$ precision matrix $\hat{\Omega}$ is evenly divided into $l = p/k$ $(\geq q)$ column blocks, e.g., $\mathbf{X}^j, 1 \leq j \leq l$. We solve a column block $\mathbf{X}^j$ at a time in the process grid. Assume the data matrix $\mathbf{A}$ has been evenly distributed into the process grid and $\vec{\mathbf{A}}_{ij}$ is the data on the ij-th core, i.e., $\mathbf{A}$ is colletion of $\vec{\mathbf{A}}_{ij}$ under a mapping scheme, which we will discuss later. Figure 8.1(b) illustrates that the 2×2 process grid is computing the first column block $\mathbf{X}^1$ while the second column block $\mathbf{X}^2$ is waiting in queues (red lines), assuming $\mathbf{X}^1, \mathbf{X}^2$ are distributed into the process grid in the same way as $\mathbf{A}$ and $\vec{\mathbf{X}}^1_{ij}$ is the block of $\mathbf{X}^1$ assigned to the ij-th core.

A typical issue in parallel computation is load imbalance, which is mainly caused by the computational disparity among cores and leads to unsatisfactory speedups. Since each step in CLIME-ADMM are basic operations like matrix multiplication, the distribution of sub-matrices over processes has a major impact on the load balance and scalability. The following discussion focuses on the matrix multiplication in the step 6 in Algorithm 7. Other steps can be easily incorporated into the framework. The matrix multiplication $\mathbf{U} = \mathbf{A}(\mathbf{A}'\mathbf{X}^1)$ can be decomposed into two steps, i.e., $\mathbf{W} = \mathbf{A}'\mathbf{X}^1$ and $\mathbf{U} = \mathbf{A}\mathbf{W}$, where $\mathbf{A} \in \Re^{n \times p}$, $\mathbf{X}^1 \in \Re^{p \times k}$, $\mathbf{W} \in$



X^5 X^6 X^7 X^8
X^1 X^2 X^3 X^4
A
$\vec{X}^2_{11}$ $\vec{X}^1_{11}$ $\vec{A}_{11}$ $\vec{A}_{12}$ $\vec{X}^1_{12}$ $\vec{X}^2_{22}$
$\vec{X}^2_{21}$ $\vec{X}^1_{21}$ $\vec{A}_{21}$ $\vec{A}_{22}$ $\vec{X}^1_{22}$ $\vec{X}^2_{22}$
Parallel IO
A_{11} A_{12} A_{13} A_{14}
A_{21} A_{22} A_{23} A_{24}
A_{31} A_{32} A_{33} A_{34}
A_{41} A_{42} A_{43} A_{44}
A_{51} A_{52} A_{53} A_{54}
A_{61} A_{62} A_{63} A_{64}

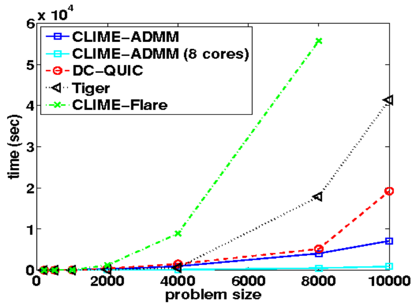
(a) Shared-Memory (b) Distributed-Memory (c) Block Cyclic

Figure 8.1: CLIME-ADMM on shared-memory and distribtued-memory architectures.

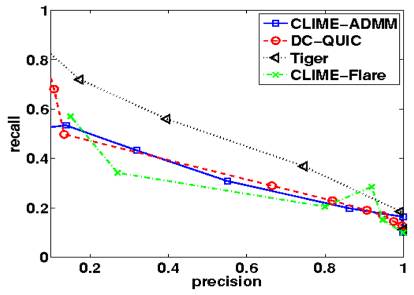
$\Re^{n\times k}$ and $\mathbf{U} \in \Re^{n\times k}$. Dividing matrices $\mathbf{A}, \mathbf{X}$ evenly into $r \times c$ large consecutive blocks like [154] will lead to load imbalance. First, since the sparse structure of $\mathbf{X}$ changes over time (Section 3.1), large consecutive blocks may assign dense blocks to some processes and sparse blocks to the other processes. Second, there will be no blocks in some processes after the multiplication using large blocks since $\mathbf{W}$ is a small matrix compared to $\mathbf{A}, \mathbf{X}$, e.g., p could be millions and n, k are hundreds. Third, large blocks may not be fit in the cache, leading to cache misses. Therefore, we use block cyclic data distribution which uses a small nonconsecutive blocks and thus can largely achieve load balance and scalability. A matrix is first divided into consecutive blocks of size $p_b \times n_b$. Then blocks are distributed into the process grid cyclically. Figure 8.1(c) illustrates how to distribute the matrix to a 2×2 process grid. $\mathbf{A}$ is divided into 3×2 consecutive blocks, where each block is of size $p_b \times n_b$. Blocks of the same color will be assigned to the same process. Green blocks will be assigned to the upper left process, i.e., $\vec{\mathbf{A}}_{11} = \{\mathbf{a}_{11}, \mathbf{a}_{13}, \mathbf{a}_{31}, \mathbf{a}_{33}, \mathbf{a}_{51}, \mathbf{a}_{53}\}$ in Figure 8.1(b). The distribution of $\mathbf{X}^1$ can be done in a similar way except the block size should be $p_b \times k_b$, where p_b is to guarantee that matrix multiplication $\mathbf{A}'\mathbf{X}^1$ works. In particular, we denote $p_b \times n_b \times k_b$ as the block size for matrix multiplication. To distribute the data in a block cyclic manner, we use a parallel I/O scheme, where processes can access the data in parallel and only read/write the assigned blocks.

8.5 Experimental Results

In this section, we present experimental results to compare CLIME-ADMM with existing algorithms and show its scalability. In all experiments, we use the low rank property of the



(a) Runtime



(b) Precision and recall

Figure 8.2: Synthetic datasets

sample covariance matrix and do not assume any other special structures. Our algorithm is implemented in a shared-memory architecture using OpenMP (http://openmp.org/wp/) and a distributed-memory architecture using OpenMPI[1] and ScaLAPACK [16][2] .

8.5.1 Comparision with Existing Algorithms

We compare CLIME-ADMM with three other methods for estimating the inverse covariance matrix, including CLIME, Tiger in package flare[3] and divide and conquer QUIC (DC-QUIC) [88]. The comparisons are run on an Intel Zeon E5540 2.83GHz CPU with 32GB main memory.

We test the efficiency of the above methods on both synthetic and real datasets. For synthetic datasets, we generate the underlying graphs with random nonzero pattern by the same way as in [89]. We control the sparsity of the underlying graph to be 0.05, and generate random graphs with various dimension. Since each estimator has different parameters to control the sparsity, we set them individually to recover the graph with sparsity 0.05, and compare the time to get the solution. The column block size k for CLIME-ADMM is 100. Figure 8.2(a) shows that CLIME-ADMM is the most scalable estimator for large graphs. We compare the precision and recall for different methods on recovering the groud truth graph structure. We run each method using different parameters (which controls the sparsity of the solution), and plot the precision and recall for each solution in Figure 8.2(b). As Tiger is parameter tuning free and achieves the

[1] http://www.open-mpi.org
[2] http://www.netlib.org/scalapack/
[3] The interior point method in [23] is written in R and extremely slow. Therefore, we use flare which is implemented in C with R interface. http://cran.r-project.org/web/packages/flare/index.html

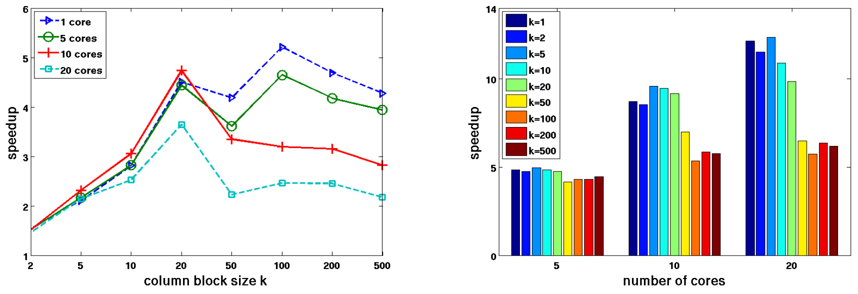
minimax optimal rate [120], it achieves the best performance in terms of recall. The other three methods have the similar performance. CLIME can also be free of parameter tuning and achieve the optimal minimax rate by solving an additional linear program which is similar to (9.5) [22]. We refer the readers to [23, 22, 120] for detailed comparisons between the two models CLIME and Tiger, which is not the focus of this paper.

We further test the efficiency of the above algorithms on two real datasets, Leukemia and Climate (see Table 1). Leukemia is gene expression data provided by [72], and the pre-processing was done by [111]. Climate dataset is the temperature data in year 2001 recorded by NCEP/NCAR Reanalysis data[4] and preprocessed by [88]. Since the ground truth for real datasets are unknown, we test the time taken for each method to recover graphs with 0.1 and 0.01 sparsity. The results are presented in Table 1. Although Tiger is faster than CLIME-ADMM on small dimensional dataset Leukemia, it does not scale well on the high dimensional dataset as CLIME-ADMM, which is mainly due to the fact that ADMM is not competitive with other methods on small problems but has superior scalability on big datasets [19]. DC-QUIC runs faster than other methods for small sparsity but dramatically slows down when sparsity increases. DC-QUIC essentially works on a block-diagonal matrix by thresholding the off-diagonal elements of the sample covariance matrix. A small sparsity generally leads to small diagonal blocks, which helps DC-QUIC to make a giant leap forward in the computation. A block-diagonal structure in the sample covariance matrix can be easily incorporated into the matrix multiplication in CLIME-ADMM to achieve a sharp computational gain. On a single core, CLIME-ADMM is faster than flare ADMM. We also show the results of CLIME-ADMM on 8 cores, showing CLIME-ADMM achieves a linear speedup (more results will be seen in Section 8.5.2). Note Tiger can estimate the spase precision matrix column-by-column in parallel, while CLIME-ADMM solves CLIME in column-blocks in parallel.

8.5.2 Scalability of CLIME ADMM

We evaluate the scalability of CLIME-ADMM in a shared memory and a distributed memory architecture in terms of two kinds of speedups. The first speedup is defined as the time on 1 core T_1^{core} over q cores T_q^{core}, i.e., $S_q^{\text{core}} = T_1^{\text{core}}/T_q^{\text{core}}$. The second speedup is caused by the use of column blocks. Assume the total time for solving CLIME column-by-column ($k = 1$) is T_1^{col}, which is considered as the baseline. The speedup of solving CLIME in column block

[4] www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis.surface.html



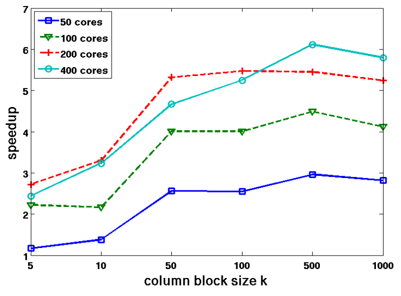
(a) Speedup S_k^{col} (b) Speedup S_q^{core}

Figure 8.3: Shared-Memory.

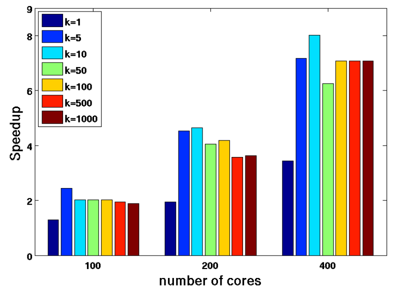
with size k over a single column is defined as $S_k^{\text{col}} = T_1^{\text{col}}/T_k^{\text{col}}$. The experiments are done on synthetic data which is generated in the same way as in Section 8.5.1. The number of samples is fixed to be $n = 200$.

Shared-memory We estimate a precision matrix with $p = 10^4$ dimensions on a server with 20 cores and 64G memory. We use OpenMP to parallelize column blocks. We run the algorithm on different number of cores $q = 1, 5, 10, 20$, and with different column block size k. The speedup S_k^{col} is plotted in Figure 8.3(a), which shows the results on three different number of cores. When $k \leq 20$, the speedups keep increasing with increasing number of columns k in each block. For $k \geq 20$, the speedups are maintained on 1 core and 5 cores, but decreases on 10 and 20 cores. The total number of columns in the shared-memory is $k \times q$. For a fixed k, more columns are involved in the computation when more cores are used, leading to more memory consumption and competition for the usage of shared cache. The speedup S_q^{core} is plotted in Figure 8.3(b), where T_1^{core} is the time on a single core. The ideal linear speedups are archived on 5 cores for all block sizes k. On 10 cores, while small and medium column block sizes can maintain the ideal linear speedups, the large column block sizes fail to scale linearly. The failure to achieve a linear speedup propagate to small and medium column block sizes on 20 cores, although their speedups are larger than large column block size. As more and more column blocks are participating in the computation, the speed-ups decrease possibly because of the competition for resources (e.g., L2 cache) in the shared-memory environment.

Distributed-memory We estimate a precision matrix with one million dimensions ($p = 10^6$), which contains one trillion parameters ($p^2 = 10^{12}$). The experiments are run on a cluster



(a) Speedup S_k^{col}



(b) Speedup S_q^{core}

Figure 8.4: Distributed-Memory.

with 400 computing nodes. We use 1 core per node to avoid the competition for the resources as we observed in the shared-memory case. For q cores, we use the process grid $\frac{q}{2} \times 2$ since $p \gg n$. The block size $p_b \times n_b \times k_b$ for matrix multiplication is $10 \times 10 \times 1$ for $k \leq 10$ and $10 \times 10 \times 10$ for $k > 10$. Since the column block CLIME problems are totally independent, we report the speedups on solving a single column block. The speedup S_k^{col} is plotted in Figure 8.4(a), where the speedups are larger and more stable than that in the shared-memory environment. The speedup keeps increasing before arriving at a certain number as column block size increases. For any column block size, the speedup also increases as the number of cores increases. The speedup S_q^{core} is plotted in Figure 8.4(b), where T_1^{core} is the time on 50 cores. A single column ($k = 1$) fails to achieve linear speedups when hundreds of cores are used. However, if using a column block $k > 1$, the ideal linear speedups are achieved with increasing number of cores. Note that due to distributed memory, the larger column block sizes also scale linearly, unlike in the shared memory setting, where the speedups were limited due to resource sharing. As we have seen, k depends on the size of process grid, block size in matrix multiplication, cache size and probably the sparsity pattern of matrices. In Table 2, we compare the performance of 1 core per node to that of using 4 cores per node, which mixes the effects of shared-memory and distributed-memory architectures. For small column block size ($k = 1, 5$), the use of multiple cores in a node is almost two times slower than the use of a single core in a node. For other column block sizes, it is still 30% slower. Finally, we ran CLIME-ADMM on 400 cores with one node per core and block size $k = 500$, and the entire computation took about 11 hours.

Table 8.1: Comparison of runtime (sec) on real datasets.

Dataset	sparsity	CLIME-ADMM		DC-QUIC	Tiger	flare CLIME
		1 core	8 cores			
Leukemia	0.1	48.64	6.27	93.88	34.56	142.5
(1255×72)	0.01	44.98	5.83	21.59	17.10	87.60
Climate	0.1	4.76 hours	0.6 hours	10.51 hours	>1 day	>1 day
(10512×1464)	0.01	4.46 hours	0.56 hours	2.12 hours	>1 day	>1 day

Table 8.2: Effect (runtime (sec)) of using different number of cores in a node with $p = 10^6$. Using one core per node is the most efficient as there is no resource sharing with other cores.

node $\times$core	k = 1	k = 5	k = 10	k = 50	k = 100	k = 500	k = 1000
100×1	0.56	1.26	2.59	6.98	13.97	62.35	136.96
25×4	1.02	2.40	3.42	8.25	16.44	84.08	180.89
200×1	0.37	0.68	1.12	3.48	6.76	33.95	70.59
50×4	0.74	1.44	2.33	4.49	8.33	48.20	103.87

Appendix

8.A Optimization Convergence Rate for CLIME ADMM

All norms in this section are defined elementwise. To recap, we solve the following problem:

$$\min \|\mathbf{X}\|_1 \quad \text{s.t.} \quad \|\mathbf{Z} - \mathbf{E}\|_\infty \leq \lambda, \mathbf{C}\mathbf{X} = \mathbf{Z} \ . \tag{8.13}$$

The Lagrangian of (8.13) is

$$L(\mathbf{X}, \mathbf{Z}, \mathbf{Y}) = \|\mathbf{X}\|_1 + \rho\langle \mathbf{Y}, \mathbf{C}\mathbf{X} - \mathbf{Z}\rangle \ , \tag{8.14}$$

where $\|\mathbf{Z} - \mathbf{E}\|_\infty \leq \lambda$. Assume that $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ satisfies the KKT conditions of (8.14), i.e.,

$$-\rho\mathbf{C}^T\mathbf{Y}^* \in \partial\|\mathbf{X}^*\|_1 \ , \tag{8.15}$$

$$\langle \mathbf{Y}^*, \mathbf{Z}^* - \mathbf{Z}\rangle \geq 0 \ , \tag{8.16}$$

$$\mathbf{C}\mathbf{X}^* = \mathbf{Z}^* \ . \tag{8.17}$$

where (8.16) holds for any $\mathbf{Z}$ satisfying $\|\mathbf{Z}-\mathbf{E}\|_\infty \le \lambda$. $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ is an optimal solution, which has the following property.

Lemma 25 *Let* $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ *be generated by ADMM and* $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ *be a KKT point. We have*

$$\|\mathbf{X}^*\|_1 - \|\mathbf{X}^{t+1}\|_1 \le \rho\langle \mathbf{Y}^*, \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\rangle \ . \tag{8.18}$$

Proof: Assume $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ is a KKT point. Using the convexity of ℓ_1 norm and (8.15), we have

$$\|\mathbf{X}^*\|_1 - \|\mathbf{X}^{t+1}\|_1 \le -\rho\langle \mathbf{C}\mathbf{Y}^*, \mathbf{X}^* - \mathbf{X}^{t+1}\rangle = -\rho\langle \mathbf{Y}^*, \mathbf{C}(\mathbf{X}^* - \mathbf{X}^{t+1})\rangle \ . \tag{8.19}$$

Setting $\mathbf{Z} = \mathbf{Z}^{t+1}$ in (8.16) yields

$$0 \le \langle \mathbf{Y}^*, \mathbf{Z}^* - \mathbf{Z}^{t+1}\rangle \ . \tag{8.20}$$

Multiplying by ρ and adding to (8.19) complete the proof. ■

In CLIME ADMM, we have the following iterates:

$$\mathbf{X}^{t+1} = \text{argmin}_{\mathbf{X}} \|\mathbf{X}\|_1 + \eta\langle \mathbf{V}^t, \mathbf{X}\rangle + \frac{\eta}{2}\|\mathbf{X} - \mathbf{X}^t\|_2^2 \ , \tag{8.21}$$

$$\mathbf{Z}^{t+1} = \underset{\|\mathbf{Z}-\mathbf{E}\|_\infty \le \lambda}{\text{argmin}} \frac{\rho}{2}\|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z} + \mathbf{Y}^t\|_2^2 \ , \tag{8.22}$$

$$\mathbf{Y}^{t+1} = \mathbf{Y}^t + \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1} \ . \tag{8.23}$$

where $\mathbf{V}^t = \frac{\rho}{\eta}\mathbf{C}(\mathbf{Y}^t + \mathbf{C}\mathbf{X}^t - \mathbf{Z}^t)$.

Throughout the proof of convergence rate, we need the following lemma.

Lemma 26 *Let* $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ *be matrices of the same size. The following equalities hold:*

$$\langle \mathbf{A} - \mathbf{B}, \mathbf{B} - \mathbf{C}\rangle = \frac{1}{2}(\|\mathbf{A} - \mathbf{C}\|_2^2 - \|\mathbf{A} - \mathbf{B}\|_2^2 - \|\mathbf{B} - \mathbf{C}\|_2^2) \ . \tag{8.24}$$

$$\langle \mathbf{A} - \mathbf{B}, \mathbf{C} - \mathbf{D}\rangle = \frac{1}{2}(\|\mathbf{D} - \mathbf{A}\|_2^2 - \|\mathbf{D} - \mathbf{B}\|_2^2 + \|\mathbf{C} - \mathbf{B}\|_2^2 - \|\mathbf{C} - \mathbf{A}\|_2^2) \ . \tag{8.25}$$

8.A.1 $O(1/T)$ Convergence Rate for Objective Function

In this section, we establish the iteration complexity for inexact ADMM (8.21)-(8.23). We begin with the following lemma for the $\mathbf{X}$ update (8.21).

Lemma 27 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.21)-(8.23). For any $\mathbf{X}$, we have*

$$\begin{aligned}\|\mathbf{X}^{t+1}\|_1 - \|\mathbf{X}\|_1 \leq -\rho\langle \mathbf{Y}^{t+1}, \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle + \frac{\rho}{2}(\|\mathbf{C}\mathbf{X} - \mathbf{Z}^t\|_2^2 - \|\mathbf{C}\mathbf{X} - \mathbf{Z}^{t+1}\|_2^2 \\ + \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) \\ + \frac{1}{2}(\|\mathbf{X} - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X} - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \, .\end{aligned} \tag{8.26}$$

Proof: Let $\partial\|\mathbf{X}^{t+1}\|_1$ be the subgradient of $\|\mathbf{X}^{t+1}\|_1$. Since $\mathbf{X}^{t+1}$ is a minimizer of (8.21), we have

$$\mathbf{0} \in \partial\|\mathbf{X}^{t+1}\|_1 + \eta(\mathbf{V}^t + \mathbf{X}^{t+1} - \mathbf{X}^t) \, . \tag{8.27}$$

Rearranging the terms gives $-\eta(\mathbf{V}^t + \mathbf{X}^{t+1} - \mathbf{X}^t) \in \partial\|\mathbf{X}^{t+1}\|_1$. Using the convexity of ℓ_1 norm, we have

$$\begin{aligned}&\|\mathbf{X}^{t+1}\|_1 - \|\mathbf{X}\|_1 \leq -\eta\langle \mathbf{V}^t + \mathbf{X}^{t+1} - \mathbf{X}^t, \mathbf{X}^{t+1} - \mathbf{X}\rangle \\ &\leq -\rho\langle \mathbf{C}(\mathbf{Y}^t + \mathbf{C}\mathbf{X}^t - \mathbf{Z}^t), \mathbf{X}^{t+1} - \mathbf{X}\rangle - \eta\langle \mathbf{X}^{t+1} - \mathbf{X}^t, \mathbf{X}^{t+1} - \mathbf{X}\rangle &(8.28)\\ &\leq -\rho\langle \mathbf{Y}^t + \mathbf{C}\mathbf{X}^t - \mathbf{Z}^t, \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle - \eta\langle \mathbf{X}^{t+1} - \mathbf{X}^t, \mathbf{X}^{t+1} - \mathbf{X}\rangle \\ &= -\rho\langle \mathbf{Y}^{t+1}, \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle - \rho\langle \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1}), \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle \\ &+ \rho\langle \mathbf{Z}^t - \mathbf{Z}^{t+1}, \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle - \eta\langle \mathbf{X}^{t+1} - \mathbf{X}^t, \mathbf{X}^{t+1} - \mathbf{X}\rangle \, . &(8.29)\end{aligned}$$

where the last equality uses (8.23). Using (8.24), the second term can be written as

$$\begin{aligned}&-\langle \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1}), \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle \\ &= -\frac{1}{2}(\|\mathbf{C}(\mathbf{X} - \mathbf{X}^t)\|_2^2 - \|\mathbf{C}(\mathbf{X} - \mathbf{X}^{t+1})\|_2^2 - \|\mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\|_2^2) \, .\end{aligned} \tag{8.30}$$

Note $\|\mathbf{C}(\mathbf{X} - \mathbf{X}^t)\|_2^2 = \|\mathbf{X} - \mathbf{X}^t\|_{\mathbf{C}^2}^2$. Using (8.25), the third term of (8.29) can be written as

$$\begin{aligned}&\langle \mathbf{Z}^t - \mathbf{Z}^{t+1}, \mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X})\rangle \\ &= \frac{1}{2}(\|\mathbf{C}\mathbf{X} - \mathbf{Z}^t\|_2^2 - \|\mathbf{C}\mathbf{X} - \mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) \, .\end{aligned} \tag{8.31}$$

Applying (8.24) on the last term of (8.29) gives

$$-\langle \mathbf{X}^{t+1} - \mathbf{X}^t, \mathbf{X}^{t+1} - \mathbf{X}\rangle = \frac{1}{2}(\|\mathbf{X} - \mathbf{X}^t\|_2^2 - \|\mathbf{X} - \mathbf{X}^{t+1}\|_2^2 - \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_2^2) \, . \tag{8.32}$$

Substituting (8.30)-(8.32) into (8.29) and rearraning the terms complete the proof. ■

The $\mathbf{Z}$ update (8.22) has the following lemma.

Lemma 28 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.21)-(8.23). For any $\mathbf{Z}$ satisfying $\|\mathbf{Z}-\mathbf{E}\|_\infty \leq \lambda$,*

$$0 \leq -\langle \mathbf{Y}^{t+1}, \mathbf{Z} - \mathbf{Z}^{t+1} \rangle \,. \tag{8.33}$$

Proof: Since $\mathbf{Z}^{t+1}$ is a minimizer of (8.22), for any $\mathbf{Z}$ satisfying the infinity norm constraint, then

$$-\langle \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1} + \mathbf{Y}^t, \mathbf{Z} - \mathbf{Z}^{t+1} \rangle \geq 0 \,. \tag{8.34}$$

Using (8.23) completes the proof. ■

Combining the results in Lemma 27 and 28 yields the $O(1/T)$ convergence rate for the objective of inexact ADMM (8.21)-(8.23).

Theorem 31 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.22)-(8.23) and $\bar{\mathbf{X}}^T = \frac{1}{T}\sum_{t=1}^T \mathbf{X}^t$. Assume $\mathbf{X}^0 = \mathbf{Z}^0 = \mathbf{Y}^0 = \mathbf{0}$ and $\eta \geq \lambda_{\max}^2(\mathbf{C})$. For any $\mathbf{C}\mathbf{X} = \mathbf{Z}$, we have*

$$\|\bar{\mathbf{X}}^T\|_1 - \|\mathbf{X}\|_1 \leq \frac{\eta\|\mathbf{X}\|_2^2}{2T} \,. \tag{8.35}$$

Proof: Assume $\mathbf{C}\mathbf{X} = \mathbf{Z}$. Multiplying (8.33) by ρ and adding (8.26) yields

$$\begin{aligned}
&\|\mathbf{X}^{t+1}\|_1 - \|\mathbf{X}\|_1 \\
&\leq -\rho\langle \mathbf{Y}^{t+1}, \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\rangle + \frac{1}{2}(\|\mathbf{Z}-\mathbf{Z}^t\|_2^2 - \|\mathbf{Z}-\mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 \\
&- \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) + \frac{1}{2}(\|\mathbf{X}-\mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}-\mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^{t+1}-\mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \,.
\end{aligned} \tag{8.36}$$

Using (8.23), the first term can be written as

$$\begin{aligned}
&-\langle \mathbf{Y}^{t+1}, \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\rangle = -\langle \mathbf{Y}^{t+1}, \mathbf{Y}^{t+1} - \mathbf{Y}^t\rangle \\
&= \frac{1}{2}(\|\mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^{t+1}\|_2^2 - \|\mathbf{Y}^{t+1} - \mathbf{Y}^t\|_2^2) \\
&= \frac{1}{2}(\|\mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2) \,.
\end{aligned} \tag{8.37}$$

Substituting back into (8.36) gives

$$\|\mathbf{X}^{t+1}\|_1 - \|\mathbf{X}\|_1 \leq \frac{\rho}{2}(\|\mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^{t+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{Z}-\mathbf{Z}^t\|_2^2 - \|\mathbf{Z}-\mathbf{Z}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2)$$

$$+\frac{1}{2}(\|\mathbf{X}-\mathbf{X}^t\|^2_{\eta\mathbf{I}-\rho\mathbf{C}^2}-\|\mathbf{X}-\mathbf{X}^{t+1}\|^2_{\eta\mathbf{I}-\rho\mathbf{C}^2}-\|\mathbf{X}^{t+1}-\mathbf{X}^t\|^2_{\eta\mathbf{I}-\rho\mathbf{C}^2})\,. \tag{8.38}$$

Assuming $\eta \geq \lambda^2_{\max}(\mathbf{C})$, $\eta\mathbf{I}-\rho\mathbf{C}^2$ is positive semidefinite. Summing over t from 0 to $T-1$ and ignoring some negative terms, we have the following telescoping sum

$$\begin{aligned}\sum_{t=0}^{T-1}\|\mathbf{X}^{t+1}\|_1-\|\mathbf{X}\|_1 &\leq \frac{\rho}{2}\|\mathbf{Y}^0\|_2^2+\frac{\rho}{2}\|\mathbf{Z}-\mathbf{Z}^0\|_2^2+\frac{1}{2}\|\mathbf{X}-\mathbf{X}^0\|^2_{\eta\mathbf{I}-\rho\mathbf{C}^2}\\ &=\frac{\rho}{2}\|\mathbf{Z}\|_2^2+\frac{1}{2}\|\mathbf{X}\|^2_{\eta\mathbf{I}-\rho\mathbf{C}^2}\\ &=\frac{\eta}{2}\|\mathbf{X}\|_2^2\,.\end{aligned} \tag{8.39}$$

where the first equality is due to $\mathbf{X}^0=\mathbf{Z}^0=\mathbf{Y}^0=\mathbf{0}$ and the second equality uses $\mathbf{C}\mathbf{X}=\mathbf{Z}$. Applying the Jensen's inequality on the left hand side completes the proof. ■

8.A.2 $O(1/T)$ Convergence Rate for the Optimality Conditions

For the $\mathbf{X}$ update (8.21), we have the following lemma.

Lemma 29 *Let $\{\mathbf{X}^t,\mathbf{Z}^t,\mathbf{Y}^t\}$ be generated by (8.21)-(8.23). We have*

$$\|\mathbf{C}\mathbf{X}^{t+1}-\mathbf{Z}^t\|_2^2+\|\mathbf{X}^{t+1}-\mathbf{X}^t\|^2_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2}\leq\|\mathbf{C}\mathbf{X}^t-\mathbf{Z}^t\|_2^2+\|\mathbf{Z}^{t-1}-\mathbf{Z}^t\|_2^2+\|\mathbf{X}^t-\mathbf{X}^{t-1}\|^2_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2}\,. \tag{8.40}$$

Proof: Setting $\mathbf{X}=\mathbf{X}^t$ in (8.28) gives

$$\begin{aligned}&\|\mathbf{X}^{t+1}\|_1-\|\mathbf{X}^t\|_1\leq-\rho\langle\mathbf{Y}^t+\mathbf{C}\mathbf{X}^t-\mathbf{Z}^t,\mathbf{C}(\mathbf{X}^{t+1}-\mathbf{X}^t)\rangle-\eta\langle\mathbf{X}^{t+1}-\mathbf{X}^t,\mathbf{X}^{t+1}-\mathbf{X}^t\rangle\\ &\leq-\rho\langle\mathbf{Y}^t,\mathbf{C}(\mathbf{X}^{t+1}-\mathbf{X}^t)\rangle-\eta\|\mathbf{X}^{t+1}-\mathbf{X}^t\|_2^2\\ &+\frac{\rho}{2}(\|\mathbf{C}\mathbf{X}^t-\mathbf{Z}^t\|_2^2+\|\mathbf{C}(\mathbf{X}^{t+1}-\mathbf{X}^t)\|_2^2-\|\mathbf{C}\mathbf{X}^{t+1}-\mathbf{Z}^t\|_2^2)\,.\end{aligned} \tag{8.41}$$

At t, (8.29) becomes

$$\begin{aligned}\|\mathbf{X}^t\|_1-\|\mathbf{X}\|_1\leq&-\rho\langle\mathbf{Y}^t,\mathbf{C}(\mathbf{X}^t-\mathbf{X})\rangle-\rho\langle\mathbf{C}(\mathbf{X}^{t-1}-\mathbf{X}^t),\mathbf{C}(\mathbf{X}^t-\mathbf{X})\rangle\\ &+\rho\langle\mathbf{Z}^{t-1}-\mathbf{Z}^t,\mathbf{C}(\mathbf{X}^t-\mathbf{X})\rangle-\eta\langle\mathbf{X}^t-\mathbf{X}^{t-1},\mathbf{X}^t-\mathbf{X}\rangle\,.\end{aligned} \tag{8.42}$$

Setting $\mathbf{X}=\mathbf{X}^{t+1}$ gives

$$\|\mathbf{X}^t\|_1-\|\mathbf{X}^{t+1}\|_1\leq-\rho\langle\mathbf{Y}^t,\mathbf{C}(\mathbf{X}^t-\mathbf{X}^{t+1})\rangle-\rho\langle\mathbf{C}(\mathbf{X}^{t-1}-\mathbf{X}^t),\mathbf{C}(\mathbf{X}^t-\mathbf{X}^{t+1})\rangle$$

$$+ \rho\langle \mathbf{Z}^{t-1} - \mathbf{Z}^t, \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\rangle - \eta\langle \mathbf{X}^t - \mathbf{X}^{t-1}, \mathbf{X}^t - \mathbf{X}^{t+1}\rangle \ . \quad (8.43)$$

Using (8.24), the second term becomes

$$\begin{aligned} &- \rho\langle \mathbf{C}(\mathbf{X}^{t-1} - \mathbf{X}^t), \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\rangle \\ &= -\frac{\rho}{2}(\|\mathbf{C}(\mathbf{X}^{t-1} - \mathbf{X}^{t+1})\|_2^2 - \|\mathbf{C}(\mathbf{X}^{t-1} - \mathbf{X}^t)\|_2^2 - \|\mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\|_2^2) \ . \end{aligned} \quad (8.44)$$

Similarly, applying (8.24) on the fourth term of (8.43) gives

$$-\eta\langle \mathbf{X}^t - \mathbf{X}^{t-1}, \mathbf{X}^t - \mathbf{X}^{t+1}\rangle = \frac{\eta}{2}(\|\mathbf{X}^{t-1} - \mathbf{X}^{t+1}\|_2^2 - \|\mathbf{X}^t - \mathbf{X}^{t-1}\|_2^2 - \|\mathbf{X}^t - \mathbf{X}^{t+1}\|_2^2) \ . \quad (8.45)$$

Adding (8.44) and (8.45) together yields

$$\begin{aligned} &- \rho\langle \mathbf{C}(\mathbf{X}^{t-1} - \mathbf{X}^t), \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\rangle - \eta\langle \mathbf{X}^t - \mathbf{X}^{t-1}, \mathbf{X}^t - \mathbf{X}^{t+1}\rangle \\ &= \frac{1}{2}(\|\mathbf{X}^{t-1} - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^t - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \\ &\leq \frac{1}{2}(\|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 + \|\mathbf{X}^t - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \ , \end{aligned} \quad (8.46)$$

where the last inequality uses $\|\mathbf{A} - \mathbf{B}\|_2^2 \leq 2(\|\mathbf{A} - \mathbf{C}\|_2^2 + \|\mathbf{B} - \mathbf{C}\|_2^2)$. Using the inequality $\langle \mathbf{A}, \mathbf{B}\rangle \leq \frac{1}{2}(\|\mathbf{A}\|_2^2 + \|\mathbf{B}\|_2^2)$, the third term of (8.43) can be written as

$$\rho\langle \mathbf{Z}^{t-1} - \mathbf{Z}^t, \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\rangle \leq \frac{\rho}{2}(\|\mathbf{Z}^{t-1} - \mathbf{Z}^t\|_2^2 + \|\mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\|_2^2) \ . \quad (8.47)$$

Substituting (8.46) and (8.47) back to (8.43), we have

$$\begin{aligned} \|\mathbf{X}^t\|_1 - \|\mathbf{X}^{t+1}\|_1 \leq -\rho\langle \mathbf{Y}^t, \mathbf{C}(\mathbf{X}^t - \mathbf{X}^{t+1})\rangle + \frac{\rho}{2}\|\mathbf{Z}^{t-1} - \mathbf{Z}^t\|_2^2 \\ + \frac{1}{2}(\|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 + \eta\|\mathbf{X}^t - \mathbf{X}^{t+1}\|_2^2) \end{aligned} \quad (8.48)$$

Adding (8.41) and (8.48) together yields

$$\begin{aligned} 0 &\leq \frac{\rho}{2}(\|\mathbf{C}\mathbf{X}^t - \mathbf{Z}^t\|_2^2 + \|\mathbf{C}(\mathbf{X}^{t+1} - \mathbf{X}^t)\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) - \eta\|\mathbf{X}^{t+1} - \mathbf{X}^t\|_2^2 \\ &+ \frac{\rho}{2}\|\mathbf{Z}^{t-1} - \mathbf{Z}^t\|_2^2 + \frac{1}{2}(\|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 + \eta\|\mathbf{X}^t - \mathbf{X}^{t+1}\|_2^2) \\ &= \frac{\rho}{2}(\|\mathbf{C}\mathbf{X}^t - \mathbf{Z}^t\|_2^2 + \|\mathbf{Z}^{t-1} - \mathbf{Z}^t\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) \\ &+ \frac{1}{2}(\|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \ . \end{aligned} \quad (8.49)$$

Dividing both sides by $\frac{\rho}{2}$ and rearranging the terms complete the proof. ■

For the $\mathbf{Z}$ update (8.22), we have the following lemma.

Lemma 30 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.21)-(8.23). We have*

$$\|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{Z}^{t+1} - \mathbf{Z}^t\|_2^2 \leq \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2 \; . \tag{8.50}$$

Proof: Setting $\mathbf{Z} = \mathbf{Z}^t$ in (8.33) gives

$$0 \leq -\langle \mathbf{Y}^{t+1}, \mathbf{Z}^t - \mathbf{Z}^{t+1} \rangle \; . \tag{8.51}$$

At t, (8.33) becomes

$$0 \leq -\langle \mathbf{Y}^t, \mathbf{Z} - \mathbf{Z}^t \rangle \; . \tag{8.52}$$

Setting $\mathbf{Z} = \mathbf{Z}^{t+1}$ yields

$$0 \leq -\langle \mathbf{Y}^t, \mathbf{Z}^{t+1} - \mathbf{Z}^t \rangle \; . \tag{8.53}$$

Adding (8.51) and (8.53) yields

$$\begin{aligned} 0 &\leq \langle \mathbf{Y}^{t+1} - \mathbf{Y}^t, \mathbf{Z}^{t+1} - \mathbf{Z}^t \rangle = \langle \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}, \mathbf{Z}^{t+1} - \mathbf{Z}^t \rangle \\ &= \frac{1}{2}(\|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 - \|\mathbf{Z}^{t+1} - \mathbf{Z}^t\|_2^2) \; . \end{aligned} \tag{8.54}$$

Rearranging the terms complete the proof. ■

Define $R_1(t+1)$ as follows:

$$R_1(t+1) = \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{Z}^{t+1} - \mathbf{Z}^t\|_2^2 + \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\frac{\eta}{\rho}\mathbf{I} - \mathbf{C}^2}^2 \; . \tag{8.55}$$

We now show that $R_1(t)$ is non-increasing by combining the results in Lemma 29 and 30 .

Lemma 31 *Let $R_1(t)$ be defined in (8.55). We have*

$$R_1(t+1) \leq R_1(t) \; . \tag{8.56}$$

Proof: Adding (8.40) and (8.50) yields

$$\begin{aligned} &\|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{Z}^{t+1} - \mathbf{Z}^t\|_2^2 + \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\frac{\eta}{\rho}\mathbf{I} - \mathbf{C}^2}^2 \\ &\leq \|\mathbf{C}\mathbf{X}^t - \mathbf{Z}^t\|_2^2 + \|\mathbf{Z}^{t-1} - \mathbf{Z}^t\|_2^2 + \|\mathbf{X}^t - \mathbf{X}^{t-1}\|_{\frac{\eta}{\rho}\mathbf{I} - \mathbf{C}^2}^2 \; . \end{aligned} \tag{8.57}$$

(8.56) follows from the definition of R_1 in (8.55). ■

Lemma 32 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.21)-(8.23) and $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ be a KKT point. We have*

$$\begin{aligned} R_1(t+1) \leq \|\mathbf{Y}^* - \mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^* - \mathbf{Y}^{t+1}\|_2^2 + \|\mathbf{Z}^* - \mathbf{Z}^t\|_2^2 - \|\mathbf{Z}^* - \mathbf{Z}^{t+1}\|_2^2 \\ + \|\mathbf{X}^* - \mathbf{X}^t\|_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2}^2 - \|\mathbf{X}^* - \mathbf{X}^{t+1}\|_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2}^2 \,. \end{aligned} \tag{8.58}$$

where $R_1(t+1)$ is defined in (8.55).

Proof: Adding (8.36) and (8.18) yields

$$\begin{aligned} 0 \leq \rho\langle \mathbf{Y}^* - \mathbf{Y}^{t+1}, \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\rangle + \frac{\rho}{2}(\|\mathbf{Z}^* - \mathbf{Z}^t\|_2^2 - \|\mathbf{Z}^* - \mathbf{Z}^{t+1}\|_2^2 + \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2 \\ - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) + \frac{1}{2}(\|\mathbf{X}^* - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^* - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \,. \end{aligned} \tag{8.59}$$

Using (8.23) and applying (8.24) on the first term, we have

$$\begin{aligned} &\langle \mathbf{Y}^* - \mathbf{Y}^{t+1}, \mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\rangle = \langle \mathbf{Y}^* - \mathbf{Y}^{t+1}, \mathbf{Y}^{t+1} - \mathbf{Y}^t\rangle \\ &= \frac{1}{2}(\|\mathbf{Y}^* - \mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^* - \mathbf{Y}^{t+1}\|_2^2 - \|\mathbf{Y}^{t+1} - \mathbf{Y}^t\|_2^2) \\ &= \frac{1}{2}(\|\mathbf{Y}^* - \mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^* - \mathbf{Y}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^{t+1}\|_2^2) \,. \end{aligned} \tag{8.60}$$

Plugging into (8.59) yields

$$\begin{aligned} 0 \leq \frac{\rho}{2}(\|\mathbf{Y}^* - \mathbf{Y}^t\|_2^2 - \|\mathbf{Y}^* - \mathbf{Y}^{t+1}\|_2^2) + \frac{\rho}{2}(\|\mathbf{Z}^* - \mathbf{Z}^t\|_2^2 - \|\mathbf{Z}^* - \mathbf{Z}^{t+1}\|_2^2 - \|\mathbf{C}\mathbf{X}^{t+1} - \mathbf{Z}^t\|_2^2) \\ + \frac{1}{2}(\|\mathbf{X}^* - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^* - \mathbf{X}^{t+1}\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2 - \|\mathbf{X}^{t+1} - \mathbf{X}^t\|_{\eta\mathbf{I}-\rho\mathbf{C}^2}^2) \,. \end{aligned} \tag{8.61}$$

Dividing both sides by $\frac{\rho}{2}$ and rearraning the terms, we have (8.58) by using (8.50) and the definition of $R_1(t)$ in (8.55). ∎

Theorem 32 *Let $\{\mathbf{X}^t, \mathbf{Z}^t, \mathbf{Y}^t\}$ be generated by (8.21)-(8.23) and $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{Y}^*\}$ be a KKT point. Assume $\mathbf{X}^0 = \mathbf{Z}^0 = \mathbf{Y}^0 = \mathbf{0}$ and $\eta \geq \lambda_{\max}^2(\mathbf{C})$. We have*

$$R_1(T) \leq \frac{\|\mathbf{Y}^*\|_2^2 + \frac{\eta}{\rho}\|\mathbf{X}^*\|_2^2}{T} \,, \tag{8.62}$$

where $R_1(T)$ is defined in (8.55).

Proof: Summing (8.58) over t from 0 to $T-1$ and igonoring some negative terms yield

$$\begin{aligned}\sum_{t=0}^{T-1} R_1(t+1) &\leq \|\mathbf{Y}^* - \mathbf{Y}^0\|_2^2 + \|\mathbf{Z}^* - \mathbf{Z}^0\|_2^2 + \|\mathbf{X}^* - \mathbf{X}^0\|^2_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2} \\ &= \|\mathbf{Y}^*\|_2^2 + \|\mathbf{Z}^*\|_2^2 + \|\mathbf{X}^*\|^2_{\frac{\eta}{\rho}\mathbf{I}-\mathbf{C}^2} \\ &= \|\mathbf{Y}^*\|_2^2 + \frac{\eta}{\rho}\|\mathbf{X}^*\|_2^2\,, \end{aligned} \tag{8.63}$$

where the first equality is due to $\mathbf{X}^0 = \mathbf{Z}^0 = \mathbf{Y}^0 = \mathbf{0}$ and the second equality uses $\mathbf{C}\mathbf{X}^* = \mathbf{Z}^*$. According to Lemma 31, $R_1(t)$ is non-increasing. Therefore,

$$TR_1(T) \leq \sum_{t=0}^{T} R_1(t+1)\,. \tag{8.64}$$

Dividing both sides by T completes the proof. ■

The optimality condition for (8.22) is given in Lemma 28, showing that KKT condition (8.16) is alway satisfied. The optimality conditions for (8.21) is

$$-\eta(\mathbf{V}^t + \mathbf{X}^{t+1} - \mathbf{X}^t) \in \partial\|\mathbf{X}^{t+1}\|_1\,. \tag{8.65}$$

Expanding $\mathbf{C}$ and using (8.23), it can be rewritten as

$$-\rho\mathbf{C}(\mathbf{Y}^{t+1} + \mathbf{X}^t - \mathbf{X}^{t+1} - \mathbf{Z}^t + \mathbf{Z}^{t+1}) - \eta(\mathbf{X}^{t+1} - \mathbf{X}^t) \in \partial\|\mathbf{X}^{t+1}\|_1\,. \tag{8.66}$$

If $\mathbf{X}^{t+1} = \mathbf{X}^t$ and $\mathbf{Z}^{t+1} = \mathbf{Z}^t$, the KKT condition (8.15) will be satisfied. Therefore, $R_1(T)$ defines the residuals of optimality conditions for (8.21)-(8.23). As $R_1(T) \to 0$, $\mathbf{C}\mathbf{X}^T = \mathbf{Z}^T, \mathbf{Z}^T = \mathbf{Z}^{T-1}$ and $\mathbf{X}^T = \mathbf{X}^{T-1}$ and thus the KKT conditions (8.15)-(8.17) are satisfied.

8.B Statistical Convergence Rates with Covariance Perturbation

In this section, we analyze the statistical convergence of the CLIME estimator [23] under perturbations of the sample covariance matrix. For the ease of reading, we first define some notations. Let $R_1, \cdots, R_k, \cdots, R_n \in \Re^p$ be n samples generated from a distribution with covariance matrix Σ_0 and true precision matrix Ω_0. The estimated covariance matrix is denoted as $\hat{\Sigma}$ and the corresponding estimated precision matrix is $\hat{\Omega}$. The pertubed covariance matrix is denoted as $\hat{S}$. The covariance matrix $\mathbf{C}$ in the main text can be either $\hat{\Sigma}$ or $\hat{S}$. The i-th element of R_k

is denoted as R_{ik}. For matrix, we use ij to index the ij-th element, e.g., $\hat{\Omega}_{ij}$. $\|\cdot\|_\infty$ and $\|\cdot\|_2$ denote the elementwise norm. $\|\cdot\|_{L_1}$ and $\|\cdot\|_{L_2}$ denote the matrix L_1 norm and L_2 norm. For the sake of completeness, we start with a brief review of some of the main results for CLIME.

8.B.1 CLIME Estimator: Bounds in terms of λ

For n samples $R_1, \dots, R_n \in \Re^p$, the sample covariance matrix $\hat{\Sigma}$, is computed as:

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(R_k - \bar{R})(R_k - \bar{R})^T = \frac{1}{n}\sum_{k=1}^{n} R_k R_k^T - \frac{1}{n}\bar{R}\bar{R}^T , \quad \text{where} \quad \bar{R} = \frac{1}{n}\sum_{k=1}^{n} R_k . \tag{8.67}$$

As a result, an entry of the sample covariance matrix is given by:

$$\hat{\Sigma}_{ij} = \frac{1}{n}\sum_{k=1}^{n} R_{ik}R_{jk} - \frac{1}{n}\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right)\left(\frac{1}{n}\sum_{k=1}^{n} R_{jk}\right) . \tag{8.68}$$

The analysis for CLIME [23] considers the following family of precision matrices:

$$\mathcal{U} = \mathcal{U}(M, q, s_0(p)) = \left\{ \Omega : \Omega \succ 0, \|\Omega\|_{L_1} \le M, \max_{1\le i\le p}\sum_{j=1}^{p} |\Omega_{ij}|^q \le s_0(p) \right\} , \tag{8.69}$$

for $0 \le q < 1$. Then, the CLIME estimator has the following guarantees:

Theorem 33 *Let $\Omega_0 \in \mathcal{U}(M, q, s_0(p))$. If $\lambda \ge \|\Omega_0\|_{L_1} \max_{ij} |\hat{\Sigma}_{ij} - \Sigma_{0,ij}|$, then we have*

$$\|\hat{\Omega} - \Omega_0\|_\infty \le 4\|\Omega_0\|_{L_1}\lambda , \tag{8.70}$$

$$\|\hat{\Omega} - \Omega_0\|_{L_2} \le c s_0(p)(4\|\Omega_0\|_{L_1})^{1-q}\lambda^{1-q} , \tag{8.71}$$

$$\frac{1}{p}\|\hat{\Omega} - \Omega_0\|_2^2 \le c s_0(p)(4\|\Omega_0\|_{L_1})^{2-q}\lambda^{2-q} , \tag{8.72}$$

where $c \le 2(1 + 2^{1-q} + 3^{1-q})$ is a constant.

Note that the deterministic bounds in Theorem 33 for precision estimation relies on $\|\hat{\Sigma} - \Sigma_0\|_\infty = \max_{i,j} |\hat{\Sigma}_{ij} - \Sigma_{0,ij}|$. In the next subsection, we establish tail bounds for the scenario where we (intentionally) perturb each entry of the sample covariance matrix, i.e., we work with $\hat{S}_{ij} = \hat{\Sigma}_{ij} + \Delta_{ij}$ where Δ_{ij} has a sub-exponential tail.

8.B.2 Bounds for λ

The following two norms will play a role in our analysis: For a scalar random variable v, let

$$\|v\|_{\psi_2} = \sup_{p\geq 1} p^{-1/2}(\mathbb{E}|v|^p)^{1/p} , \quad \text{and} \quad \|v\|_{\psi_1} = \sup_{p\geq 1} p^{-1}(\mathbb{E}|v|^p)^{1/p} . \tag{8.73}$$

Then, v is called a *sub-Gaussian* random variable if $\|v\|_{\psi_2} \leq K_2$ for a constant K_2, and v is called a *sub-exponential* random variable if $\|v\|_{\psi_1} \leq K_1$ for a constant K_1. In the literature, $\|v\|_{\psi_2}$ is referred to as the *sub-Gaussian norm* and $\|v\|_{\psi_1}$ is referred to as the *sub-exponential norm*. Note that, ignoring constants, sub-exponential tails decay at $\exp(-t)$ whereas sub-Gaussian tails decay as $\exp(-t^2/2)$ so that sub-exponential tails are heavier than sub-Gaussian tails.

The following result will be used in our analysis:

Lemma 33 *Let v_i, v_j be sub-Gaussian random variables with $\max\{\|v_i\|_{\psi_2}, \|v_j\|_{\psi_2}\} \leq K_2$. Then $v_iv_j - \mathbb{E}[v_iv_j]$ is a sub-exponential random variable with $\|v_iv_j - \mathbb{E}[v_iv_j]\|_{\psi_1} \leq 4K_2^2$.*

Proof: By definition,

$$\|\mathbb{E}[v_iv_j]\|_{\psi_1} = |\mathbb{E}[v_iv_j]| \leq \mathbb{E}|v_iv_j| \leq \|v_iv_j\|_{\psi_1} . \tag{8.74}$$

Using triangle inequality, we have

$$\|v_iv_j - \mathbb{E}[v_iv_j]\|_{\psi_1} \leq \|v_iv_j\|_{\psi_1} + \|\mathbb{E}[v_iv_j]\|_{\psi_1} \leq 2\|v_iv_j\|_{\psi_1} . \tag{8.75}$$

Since v_i, v_j are sub-Gaussian random variables, for any $p \geq 1$,

$$\mathbb{E}|v_i|^p \leq (K_2\sqrt{p})^p \quad \text{and} \quad \mathbb{E}|v_j|^p \leq (K_2\sqrt{p})^p . \tag{8.76}$$

Then, using Cauchy-Schwartz inequality

$$\mathbb{E}|v_iv_j|^p = \mathbb{E}|v_i|^p|v_j|^p \leq \left(\mathbb{E}|v_i|^{2p}\mathbb{E}|v_j|^{2p}\right)^{1/2} \leq \left((K_2\sqrt{2p})^{2p}(K_2\sqrt{2p})^{2p}\right)^{1/2} = K_2^{2p}2^pp^p .$$

Hence,

$$\|v_iv_j\|_{\psi_1} = \sup_{p\geq 1} p^{-1}(\mathbb{E}|v_iv_j|^p)^{1/p} \leq 2K_2^2 .$$

The result then follows from (8.75). ■

We also need the following Bernstein-type inequality for sums of independent sub-exponential random variables [196]:

Theorem 34 *Let $v_1, \ldots, v_n$ be independent centered sub-exponential random variables, and $K_1 = \max_i \|v_i\|_{\psi_1}$. Then, for every $\mathbf{b} = (b_1, \ldots, b_n) \in R^n$ and every $t \geq 0$, we have*

$$\mathbb{P}\left\{\left|\sum_{k=1}^{n} b_k v_k\right| \geq t\right\} \leq 2\exp\left\{-c_0 \min\left(\frac{t^2}{K_1^2\|\mathbf{b}\|_2^2}, \frac{t}{K_1\|\mathbf{b}\|_\infty}\right)\right\} , \tag{8.77}$$

where $c_0 > 0$ is an absolute constant.

We will be also using the following form of the above result:

Corollary 5 *Let $v_1, \ldots, v_n$ be independent centered sub-exponential random variables, and $K_1 = \max_i \|v_i\|_{\psi_1}$. Then, for every $\epsilon \geq 0$, we have*

$$\mathbb{P}\left\{\left|\frac{1}{n}\sum_{k=1}^{n} v_k\right| \geq \epsilon\right\} \leq 2\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{K_1^2}, \frac{\epsilon}{K_1}\right) n\right\} , \tag{8.78}$$

where $c_0 > 0$ is an absolute constant.

Next, we consider perturbing the covariance matrix $\hat{\Sigma}$ using independent zero-mean sub-exponential random variables. First, we illustrate that the nature of the tail bounds stay unchanged under such perturbations. Then, we show that one can do deterministic perturbations to get coarser and/or truncated representations of the sample covariance matrix, saving on the memory foot-print of the covariance matrix without affecting the statistical guarantees.

Let Δ_{ij} be independent zero mean sub-exponential random variables, and we consider the modified covariance matrix with entries:

$$\hat{S}_{ij} = \frac{1}{n}\sum_{k=1}^{n} R_{ik}R_{jk} - \frac{1}{n}\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right)\left(\frac{1}{n}\sum_{k=1}^{n} R_{jk}\right) + \Delta_{ij} \, . \tag{8.79}$$

Then, we have the following result:

Theorem 35 *Let $K_2 = \max_i \|R_{i\cdot}\|_{\psi_2}$ and $K_1 = \max_{ij} \|\Delta_{ij}\|_{\psi_1}$. Assuming $K_1 \leq 4K_2^2$, we have*

$$\mathbb{P}\left\{\max_{ij} |\hat{S}_{ij} - \Sigma_{0,ij}| \geq \epsilon\right\} \leq 6\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{36c_1^2K_2^4}, \frac{\epsilon}{12c_1K_2^2}\right) n\right\} , \tag{8.80}$$

for suitable positive constant c_0, c_1.

Proof: By definition, for any i, j,

$$\begin{aligned}
&\mathbb{P}\left\{|\hat{S}_{ij} - \Sigma_{0,ij}| \geq \epsilon\right\} \\
&= \mathbb{P}\left\{\left|\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}R_{jk} - \Sigma_{0,ij}\right) + \Delta_{ij} - \frac{1}{n}\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right)\left(\frac{1}{n}\sum_{k=1}^{n} R_{jk}\right)\right| \geq \epsilon\right\} \\
&\leq \mathbb{P}\left\{\left|\frac{1}{n}\sum_{k=1}^{n} R_{ik}R_{jk} - \Sigma_{0,ij}\right| \geq \epsilon/3\right\} + \mathbb{P}\left\{|\Delta_{ij}| \geq \epsilon/2\right\} \\
&\qquad + \mathbb{P}\left\{\left|\frac{1}{n}\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right)\left(\frac{1}{n}\sum_{k=1}^{n} R_{jk}\right)\right| \geq \epsilon/3\right\}
\end{aligned} \tag{8.81}$$

where the last inequality follows from the union bound. Each term in the summation considers a large deviation bound for a sub-exponential random variable. For the first term, from Lemma 33, $K_{1,1} = \|R_iR_j - \mathbb{E}[R_iR_j]\|_{\psi_1} \leq 4K_2^2$. For the second term, from the assumption regarding Δ_{ij}, $K_{1,2} = \|\Delta_{ij}\|_{\psi_1} \leq 4K_2^2$. Now, we focus on the third term. Recall that the sub-Gaussian norm of the sum of sub-Gaussian random variables satisfy the following inequality [196]:

$$\left\|\sum_{k=1}^{n} R_{ik}\right\|_{\psi_2}^2 \leq c_1 \sum_{k=1}^{n} \|R_{ik}\|_{\psi_2}^2 \ , \tag{8.82}$$

for an absolute constant c_1. In our context, since $\|R_{ik}\|_{\psi_2} \leq K_2$, we have

$$\left\|\sum_{k=1}^{n} R_{ik}\right\|_{\psi_2} \leq \sqrt{c_1 n}K_2 \quad \Rightarrow \quad \left\|\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right\|_{\psi_2} \leq \sqrt{\frac{c_1}{n}}K_2 \leq \sqrt{c_1}K_2 \ . \tag{8.83}$$

From Lemma 33, we have

$$K_{1,3} = \left\|\left(\frac{1}{n}\sum_{k=1}^{n} R_{ik}\right)\left(\frac{1}{n}\sum_{k=1}^{n} R_{jk}\right)\right\|_{\psi_1} \leq 4c_1K_2^2 \ . \tag{8.84}$$

Then, considering all three terms, using Corollary 5 for the first two terms and Theorem 34 for the third term, we have

$$\begin{aligned}
&\mathbb{P}\left\{|\hat{S}_{ij} - \Sigma_{0,ij}| \geq \epsilon\right\} \\
&\leq 2\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{9K_{1,1}^2}, \frac{\epsilon}{3K_{1,1}}\right)n\right\} + 2\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{9K_{1,2}^2}, \frac{\epsilon}{3K_{1,2}}\right)n\right\} \\
&\qquad + 2\exp\left\{-c_0 \min\left(\frac{\epsilon^2 n^2}{9K_{1,3}^2}, \frac{\epsilon n}{3K_{1,3}}\right)\right\}
\end{aligned}$$

$$\leq 4\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{36K_2^4}, \frac{\epsilon}{12K_2^2}\right) n\right\} + 2\exp\left\{-c_0 \min\left(\frac{\epsilon^2 n}{36c_1^2K_2^4}, \frac{\epsilon}{12c_1K_2^2}\right) n\right\}$$
$$\leq 6\exp\left\{-c_0 \min\left(\frac{\epsilon^2}{36c_1^2K_2^4}, \frac{\epsilon}{12c_1K_2^2}\right) n\right\} .$$

That completes the proof. ■

In particular, for sufficient number of samples such that $c\sqrt{\log p/n} \leq 3c_1K_2^4$, we have

$$\mathbb{P}\left\{\max_{ij} |\hat{S}_{ij} - \Sigma_{0,ij}| \geq c\sqrt{\log p/n}\right\} \leq 6\exp\left\{-\frac{c^2c_0}{36c_1^2K_2^4}\log p\right\} \leq 6p^{-c_3} , \tag{8.85}$$

where c_3 is a suitable constant. Note that the above corresponds to the result discussed in the main text.

A special case of such perturbations arise by choosing constant Δ_{ij} for each (i, j) with $|\Delta_{ij}| \leq c\sqrt{\frac{\log p}{n}}$ in order to truncate or coarsen entries in the sample covariance matrix. In particular,

(i) if $|\hat{\Sigma}_{ij}| \leq c\sqrt{\frac{\log p}{n}}$, then it can be safely truncated to 0; and

(ii) numeric representation of any $\hat{\Sigma}_{ij}$ can be coarsened to the level $c\sqrt{\frac{\log p}{n}}$, e.g., one can rewrite

$$\hat{\Sigma}_{ij} = 1.29\underbrace{317542365}_{\leq c\sqrt{\frac{\log p}{n}}} \qquad \text{as} \qquad \hat{S}_{ij} = 1.29$$

without affecting the statistical properties of the estimated precision matrix $\hat{\Omega}$. Such truncation and coarsening can lead to significant savings in the memory foot-print of the sample covariance matrix.

Chapter 9

Gaussian Copula Precision Estimation with Missing Values

9.1 Introduction

In recent years, considerable effort [7, 60, 136, 159, 23, 22, 120, 217] has been invested in obtaining an accurate estimate of the precision matrix based on the sample covariance matrix, especially when the true precision matrix is assumed to be sparse [217]. Suitable estimators and corresponding statistical convergence rates have been established for a variety of settings, including distributions with sub-Gaussian tails, polynomial tails [159, 23, 120].

Although these sparse precision estimators are primarily designed to deal with fully observed data, recently, they have also been generalized to handle data with missing values [117, 181, 122, 121, 100], which often occur in real world applications, e.g., drop-outs of sensors in a sensor network or missing measurements of temperature or rain in climate. To deal with data with missing values, a variety of methods apply expectation maximization (EM) algorithms on imputed data, which are iterative methods but lack theoretical guarantees [117, 181]. In particular, [181] proposed an EM algorithm named MissGlasso to deal with missing values using Glasso. MissGlasso first imputes the missing values in the E-step and then solves the Glasso problem on the imputed data in the M-step. As EM converges to a local optimum, it is difficult to establish theoretical guarantees for the MissGlasso procedure. Without using the EM algorithm, [121] employed projected gradient descent to solve a sequence of regression problems or PGlasso to estimate the sparse precision matrix of incomplete data. Theoretical guarantees are also established for the PGlasso estimator. [100] introduced a simple plug-in procedure for incomplete data which simply applies existing estimators to the observed data by disregarding the missing values. Such simple plug-in estimators for missing values can leverage existing theoretical results and thus still have similar statistical guarantees, including rate of convergence and consistency. However, these sparse precision estimators rely on the Gaussian assumption, which may not be appropriate for real datasets which are usually non-Gaussian.

To deal with non-Gaussian data, [118] proposed Gaussian copula graphical models where existing estimators can be generalized to the *non-paranormal* distributions simply using one additional procedure, i.e., estimating nonparametric correlations. Non-paranormal distributions can be considered as a non-parametric extension of the normal distribution where suitable univariate monotone transformations of the covariates are jointly distributed as a multivariate Gaussian. It has also been shown that the nonparanormal is equivalent to Gaussian copula distribution [119, 194, 193]. Therefore, the estimated correlation matrix of the data after transformation can be plugged into the standard sparse precision estimators with Gaussian assumption. The plug-in procedure can leverage existing theoretical results and achieve the optimal statistical rate of convergence. A similar procedure has also been studied independently by [212]. However, whether Gaussian copula graphical models can deal with missing values and maintain the optimal statistical rate of convergence is still unknown.

In this chapter, we propose $\underline{\text{Do}}$uble $\underline{\text{P}}$lug-$\underline{\text{in}}$ $\underline{\text{G}}$aussian (DoPinG) copula estimators to deal with missing values, which estimates the sparse precision matrix corresponding to the non-paranormal distribution. DoPingG copula estimators essentially combines two plug-in procedures for dealing with missing values [100] and non-Gaussian data [118], yielding a fairly rich family of estimators to deal with incomplete data from the non-paranormal family. Such estimators consider the following three steps: (1) estimate non-parametric correlations, such as Kendall's tau and Spearman's rho, between all pairs of covariates by suitably disregarding missing values; (2) estimate the non-paranormal correlation matrix using the Kendall's tau or Spearman's rho correlation matrix; (3) plug the estimated correlation matrix into existing sparse precision estimators, e.g., graphical LASSO [7, 60], Dantzig selector [217], CLIME [23], etc.

Our analysis follows the development in [118] with one important difference: the samples we consider can have missing values. We investigate how missing values affect the accuracy of covariance estimation, and in turn precision estimation. In particular, the theoretical analysis of DoPinG copula estimators considers two probability spaces, i.e., probability over samples and probability over missing values. We assume that the data is missing completely at random (MCAR) [100], where any element is missing with probability δ. We prove that DoPinG copula estimators consistently estimate the non-paranormal correlation matrix at a rate of $O(\frac{1}{(1-\delta)}\sqrt{\frac{\log p}{n}})$.

For estimating the precision matrix, one can use any of the available estimators, such as the graphical lasso [7], graphical Dantzig selector [217], as discussed in [118, 100]. We consider the CLIME estimator [23] for our analysis. The CLIME estimator has strong statistical guarantees for consistency along with rates [23], and also comes with inherent computational advantages [203]. In particular, a large scale distributed algorithm has been developed in [203], which can scale up to millions of dimensions and trillions of parameters, using hundreds of cores. We provide experimental results to show the effect of sample size and percentage of missing data on the model performance. Experimental results show that DoPinG is significantly better than estimators like mGlasso, which are primarily designed for Gaussian data.

The rest of this chapter is organized as follows. We propose nonparanormal dual plug-in estimators with missing values in Section 9.2. In Section 9.3, we give the theoretical guarantees in terms of rates of convergences under element-wise L_∞ norm. We present experimental results in Section 9.4.

9.2 Gaussian Copula Precision Estimation with Missing Values

We consider a p-dimensional *non-paranormal* distribution [118]. For univariate monotone functions $f_1, \dots, f_p$ and a positive definite correlation matrix $\Sigma^0 \in \mathbb{R}^{p \times p}$, a p-dimensional random variable $X = (X_1, \dots, X_p)^T$ has a non-paranormal distribution $X \sim \text{NPN}_p(f, \Sigma^0)$ if $f(X) = (f_1(X_1), \dots, f_p(X_p)) \sim N_p(0, \Sigma^0)$, a p-dimensional multi-variate Gaussian distribution with correlation matrix Σ^0. We focus on estimating the sparse precision matrix $\Omega_0 = \Sigma_0^{-1}$ corresponding to the non-paranormal distribution.

Let $x_1, \dots, x_n \in \mathbb{R}^p$ be samples drawn independently from $\text{NPN}_p(f, \Sigma^0)$. We further assume that for dimension j, x_{ij} will be missing with probability $\delta \in [0, 1]$. Let $b_{ij} = 1$ if x_{ij} is observed, and $b_{ij} = 0$ otherwise. Thus, $P(b_{ij} = 1) = 1 - \delta$. We assume the data is missing completely at random (MCAR) [100].

In order to estimate the precision matrix Ω^0 using CLIME, we need an empirical estimate $\hat{S}_n$ of the correlation matrix Σ^0. In particular, the elementwise L_∞ norm between the matrices need to be suitably bounded for norm consistency of precision estimation. As shown in [118] , $\hat{S}_n$ can be efficiently computed from the empirical Kendall's tau or Spearman's rho correlation matrix. Hereafter, for ease of notation, we drop the subscript n on $\hat{S}$ and other sample estimates.

DoPinG copula estimators consider three steps in estimating the precision matrix. First, suitably generalizing the plug-in procedure for estimating non-parametric correlations to handle missing values, pairwise Kendall's tau or Spearman's rho correlation between covariates is estimated. Second, the correlation matrix corresponding to the non-paranormal distribution is estimated using the Kendall's tau or Spearman's rho correlation matrices. Third, the precision matrix is estimated by simply plugging in the estimated correlation matrix into existing sparse precision matrix estimators. We discuss each one of these steps below.

9.2.1 Kendall's tau with missing values

Given that samples have missing values, we compute the Kendall's tau for dimensions (j, k) using the n_{jk} *effective* independent samples which have values for both dimensions. In particular, we estimate Kendall's rho as:

$$\hat{\tau}_{jk} = \frac{1}{n_{jk}(n_{jk}-1)} \sum_{\substack{i,i'=1 \\ i \neq i'}}^{n} b_{ij} b_{ik} b_{i'j} b_{i'k} \text{sign}((x_i^j - x_{i'}^j)(x_i^k - x_{i'}^k)) \ , \tag{9.1}$$

where $n_{jk} = \sum_{i=1}^{n} b_{ij} b_{ik}$. Note for the i-th sample, both the j- and k-th dimensions should not be missing. In other words, the samples with missing values will not be considered in the estimation of the Kendall' tau.

The second step is to estimate the correlation matrix directly based on the Kendall's tau. Following [118, 106, 57], we consider the following estimator $\hat{S}^{\tau} = [\hat{S}^{\tau}_{jk}]$ for the estimated correlation matrix Σ^0:

$$\hat{S}^{\tau}_{jk} = \begin{cases} \sin\left(\frac{\pi}{2}\hat{\tau}_{jk}\right) & \text{if } j \neq k \\ 1 & \text{if } j = k \ . \end{cases} \tag{9.2}$$

9.2.2 Spearman's rho with missing values

Similar to the estimation of Kendall's tau for missing values, we also compute the Spearman's rho for dimensions (j, k) using the n_{jk} *effective* independent samples which have values for both dimensions. In particular, $n_{jk} = \sum_{i=1}^{n} b_{ij} b_{ik}$. Let r_i^j be the rank of x_i^j among the n_{jk}

samples with values and $\bar{r}_{jk}$ be the average, i.e., $\bar{r}_{jk} = \frac{1}{n_{jk}} \sum_{i=1}^{n} r_i^j b_{ij} b_{ik}$. Spearman's rho is defined as follows:

$$\hat{\rho}_{jk} = \frac{\sum_{i=1}^{n} (r_i^j - \bar{r}_{jk})(r_i^k - \bar{r}_{jk}) b_{ij} b_{ik}}{\sqrt{\sum_{i=1}^{n} [(r_i^j - \bar{r}_{jk})^2 b_{ij} b_{ik}] \sum_{i=1}^{n} [(r_i^k - \bar{r}_{jk})^2 b_{ij} b_{ik}]}} , \tag{9.3}$$

which is the first step in DoPinG.

Based on the estimate of the Spearman's rho (9.3), following [118, 212] , the second step is to estimate $\hat{S}^\rho = [\hat{S}^\rho_{jk}]$ for the unknown correlation matrix Σ^0:

$$\hat{S}^\rho_{jk} = \begin{cases} 2 \sin\left(\frac{\pi}{6} \hat{\rho}_{jk}\right) & \text{if } j \neq k \\ 1 & \text{if } j = k \ . \end{cases} \tag{9.4}$$

9.2.3 Plugin estimate for CLIME

Having obtained $\hat{S}$ ($\hat{S}^\tau$ or $\hat{S}^\rho$), we can plugin it into any sparse precision estimators, e.g., graphical lasso [7], graphical Dantzig selector [217], CLIME [23]. In particular, we plugin $\hat{S}$ into the CLIME estimator [212]:

$$\hat{\Omega}_n = \text{argmin}_{\hat{\Omega}} \|\hat{\Omega}\|_1 \quad \text{s.t.} \quad \|\hat{S}\hat{\Omega} - \mathbf{I}\|_\infty \leq \lambda_n \ , \tag{9.5}$$

where λ_n is a tuning parameter and $\mathbf{I}$ is an identity matrix. The CLIME estimator has strong statistical guarantees [23], and also comes with inherent computational advantages. The estimator can scale up to millions of dimensions and can be run on hundreds of cores [203]. In [203], (9.5) is decomposed into solving $\lceil p/k \rceil$ independent column block linear programs where each column block contains $k (1 \leq k \leq p)$ columns. Denoting $\mathbf{X} \in \Re^{p \times k}$ be k columns of $\hat{\Omega}$, (9.5) can be written as

$$\min \|\mathbf{P}\|_1 \quad \text{s.t.} \quad \|\hat{S}\mathbf{P} - \mathbf{E}\|_\infty \leq \lambda_n \ , \tag{9.6}$$

which can be solved by an inexact ADMM algorithm [19, 200] given in Algorithm 7 [203] where ρ, η are parameters of ADMM and

$$\textit{soft}(\mathbf{P}, \gamma) = \begin{cases} P_{ij} - \gamma \ , & \text{if } P_{ij} > \gamma \ , \\ P_{ij} + \gamma \ , & \text{if } P_{ij} < -\gamma \ , \\ 0 \ , & \text{otherwise} \end{cases}$$

Algorithm 7 Column Block Inexact ADMM for CLIME

```
1:  Input: $\hat{S}, \lambda_n, \rho, \eta$
2:  Output: $\mathbf{P}$
3:  Initialization: $\mathbf{P}^0, \mathbf{Z}^0, \mathbf{Y}^0, \mathbf{V}^0, \hat{\mathbf{V}}^0 = 0$
4:  for $t = 0$ to $T-1$ do
5:      X-update: $\mathbf{P}^{t+1} = \textit{soft}(\mathbf{P}^t - \mathbf{V}^t, \frac{1}{\eta})$, where
6:      Mat-Mul: $\mathbf{U}^{t+1} = \hat{S}\mathbf{P}^{t+1}$
7:      Z-update: $\mathbf{Z}^{t+1} = \textit{box}(\mathbf{U}^{t+1} + \mathbf{Y}^t, \lambda_n)$, where
8:      Y-update: $\mathbf{Y}^{t+1} = \mathbf{Y}^t + \mathbf{U}^{t+1} - \mathbf{Z}^{t+1}$
9:      Mat-Mul: $\hat{\mathbf{V}}^{t+1} = \hat{S}\mathbf{Y}^{t+1}$
10:     V-update: $\mathbf{V}^{t+1} = \frac{\rho}{\eta}(2\hat{\mathbf{V}}^{t+1} - \hat{\mathbf{V}}^t)$
11: end for
```

$$\textit{box}(\mathbf{P}, \mathbf{E}, \lambda_n) = \begin{cases} E_{ij} + \lambda\ , & \text{if } P_{ij} - E_{ij} > \lambda_n\ , \\ P_{ij}\ , & \text{if } |P_{ij} - E_{ij}| \le \lambda_n\ , \\ E_{ij} - \lambda\ , & \text{if } P_{ij} - E_{ij} < -\lambda_n\ , \end{cases}$$

While steps 5, 7, 8 and 10 amount to elementwise operations, the most intensive computation is matrix multiplication in steps 6 and 9 which can be solved in parallel.

Note that the estimated correlation matrix $\hat{S}$ ($\hat{S}^\tau$ or $\hat{S}^\rho$) may be not positive semi-definite. Sparse precision estimators do require the positive semi-definiteness assumption in theory and most algorithms may fail if the input correlation matrix is not positive semi-definite [118, 100]. The inexact ADMM algorithm for CLIME in Algorithm 7 does not necessarily require $\hat{S}$ to be positive semi-definite. As long as the linear programs (9.5) have solutions, Algorithm 7 still works, although there is no guarantee that the solution is positive definite. Therefore, one may project the input correlation matrix onto the cone of positive semi-definite matrix in order to obtain a positive definite precision matrix with high probability using Algorithm 7. We study the effect of the two choices on the performance of DoPinG in experiments in Section 4.

9.3 Theoretical Analysis

In this section, we present statistical guarantees for the proposed DoPinG by leveraging existing analysis in [118, 23, 212]. Note that the consistency analysis of the CLIME estimate $\hat{\Omega}$ relies on obtaining a consistent estimate of the covariance Σ^0, defined in terms of the elementwise L_∞ norm of the difference $(\hat{S} - \Sigma^0)$. Therefore, we first analyze $\sup_{jk} \left|\hat{S}^\tau_{jk} - \Sigma^0_{jk}\right|$ for the Kendall's tau ($\hat{S} = \hat{S}^\tau$) and Spearman's rho ($\hat{S} = \hat{S}^\rho$) seperately. Our proof operates on two probability spaces, i.e., probabilities over the samples $\mathbb{P}_X$ and probabilities over the Bernoulli missing values $\mathbb{P}_B$. Then, we plug the results into the consistency analysis of the CLIME to obtain the optimal statistical rate of convergence.

We first consider the probabilities over missing values in the following lemma which we need in the analysis of Kendall's tau and Spearman's rho:

Lemma 34 *Let $B = [b_{ij}] \in \{0,1\}^{n\times p}$ be an binary matrix. Assume b_{ij} is i.i.d. with a Bernoulli distribution where $P(b_{ij} = 0) = \delta$ and $P(b_{ij} = 1) = 1 - \delta$. Let $n_{jk} = \sum_{i=1}^n b_{ij}b_{ik}$. For any $m > 0$, and any $0 < \epsilon < 1$, we have*

$$\mathbb{P}_B\left(\sum_{j,k}\exp\left\{-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right) > \frac{1}{p^m}\right\}$$
$$\leq \exp\left(-(\epsilon^2(1-\delta)^2 n/2 - 2\log p)\right) \ , \tag{9.7}$$

Proof: Since n_{jk} is a sum of n independent Bernoulli random variables $b_{ij}b_{ik}$ with $P(b_{ij}b_{ik} = 1) = (1-\delta)^2$, by linearity of expectation and independence of samples, we have $E[n_{jk}] = \sum_{i=1}^n E[b_{ij}b_{ik}] = n(1-\delta)^2$. By standard Chernoff bounds, for any $\epsilon < 1$, we have

$$\mathbb{P}_B\left(n_{jk} < E[n_{jk}](1-\epsilon)\right) \leq \exp\left(-\epsilon^2(1-\delta)^2 n/2\right)$$
$$\Rightarrow \mathbb{P}_B\left(\exp\left\{-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right\} \geq \frac{1}{p^{m+2}}\right)$$
$$\leq \exp\left(-\epsilon^2(1-\delta)^2 n/2\right) \ , \tag{9.8}$$

where we have substituted the expectation $E[n_{jk}]$. By considering probabilities over the missing values, we have

$$\mathbb{P}_B\left(\sum_{j,k}\exp\left\{-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right\} > \frac{1}{p^m}\right)$$
$$\leq \sum_{j,k}\mathbb{P}_B\left(\exp\left\{-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right\} > \frac{1}{p^{m+2}}\right)$$
$$\leq p^2 \exp\left(-\epsilon^2(1-\delta)^2 n/2\right)$$

$$= \exp\left(-(\epsilon^2(1-\delta)^2 n/2 - 2\log p)\right) \ , \tag{9.9}$$

which completes the proof. ■

9.3.1 Kendall's Tau with Missing Values

The following theorem shows that $\sup_{jk} \left|\hat{S}^{\tau}_{jk} - \Sigma^0_{jk}\right| \leq O(\sqrt{\log p/n})$ with high probability.

Theorem 36 *For any $n \geq 1$, for any $m > 0$, and any $0 < \epsilon < 1$, with probability at least $(1 - \frac{1}{p^m})(1 - \exp(-(\epsilon^2(1-\delta)^2 n/2 - 2\log p))$, we have*

$$\sup_{jk} \left|\hat{S}^{\tau}_{jk} - \Sigma^0_{jk}\right| \leq \frac{\pi}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}} \ . \tag{9.10}$$

Proof: Since $\hat{\tau}_{jk}$ is an unbiased estimator of τ_{jk}, $E[\hat{\tau}_{jk}] = \tau_{jk}$. Using (9.2), we have

$$\begin{aligned}
&\mathbb{P}_X\left(\left|\hat{S}_{jk} - \Sigma^0_{jk}\right| > t\right) \\
&= \mathbb{P}_X\left(\left|\sin\left(\frac{\pi}{2}\hat{\tau}_{jk}\right) - \sin\left(\frac{\pi}{2}\tau_{jk}\right)\right| > t\right) \\
&\leq \mathbb{P}_X\left(\left|\hat{\tau}_{jk} - \tau_{jk}\right| > \frac{2}{\pi}t\right) \\
&\leq \exp\left(-\frac{n_{jk}t^2}{\pi^2}\right) \ ,
\end{aligned} \tag{9.11}$$

where the last inequality uses the Hoeffding bound for the U-statistics [118, 84]. Application of the union bound yields

$$\begin{aligned}
&\mathbb{P}_X\left(\sup_{jk}\left|\hat{S}^{\tau}_{jk} - \Sigma^0_{jk}\right| > t\right) \\
&\leq \sum_{j,k}\exp\left(-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right) \ ,
\end{aligned} \tag{9.12}$$

where we have substituted $t = \frac{\pi}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}}$. The bound in the above form is itself a random variable, and the elements of the sum are identically distributed but are not independent.

By considering probabilities over the missing values and using Lemma 34, we have

$$\begin{aligned}
&\mathbb{P}_B\left(\mathbb{P}_X\left(\sup_{jk}\left|\hat{S}^{\tau}_{jk} - \Sigma^0_{jk}\right| \leq t\right) \geq \left(1 - \frac{1}{p^m}\right)\right) \\
&\geq 1 - \exp\left(-(\epsilon^2(1-\delta)^2 n/2 - 2\log p)\right) \ .
\end{aligned} \tag{9.13}$$

Noting that the random variables (X, B) are independent completes the proof. ■

9.3.2 Spearman's Rho with Missing Values

As we work on the n_{jk} effective samples wth values by disregarding missing values, we can leverage the analysis in [118] except n_{jk} is a random variable. Following [118], (9.3) can be rewritten as [83, 118]:

$$\begin{aligned}\hat{\rho}_{jk} &= \frac{3\sum_{i=1}^{n}\sum_{s=1}^{n}\sum_{t=1}^{n}\text{sign}(x_i^j - x_s^j)(x_i^k - x_t^k)b_{ij}b_{ik}b_{sj}b_{sk}b_{tj}b_{tk}}{n_{jk}^3 - n_{jk}} \\ &= \frac{n_{jk}-2}{n_{jk}+1}U_{jk} + \frac{3}{n_{jk}+1}\hat{\tau}_{jk} \ . \end{aligned} \tag{9.14}$$

where $\hat{\tau}_{jk}$ is Kendall's tau statistics and U_{jk} is a 3rd-order U-statistics

$$U_{jk} = \frac{3\sum_{i\neq s\neq t}\text{sign}(x_i^j - x_s^j)(x_i^k - x_t^k)b_{ij}b_{ik}b_{sj}b_{sk}b_{tj}b_{tk}}{n_{jk}(n_{jk}-1)(n_{jk}-2)} \ . \tag{9.15}$$

Note $n_{jk} = \sum_{i=1}^{n} b_{ij}b_{ik}$ is a sum of n independent Bernoulli random variables $b_{ij}b_{ik}$ with $\mathbb{E}(n_{ij}) = (1-\delta)^2 n$.

Theorem 37 *For any $m > 0$, $0 < \epsilon < 1$, and*

$$n \geq \frac{36}{(m+2)(1-\epsilon)(1-\delta)^2 \log p} \ , \tag{9.16}$$

with probability at least $(1 - \frac{1}{p^m})(1 - \exp(-(\epsilon^2(1-\delta)^2 n/2 - 2\log p))$, we have

$$\sup_{jk} \left|\hat{S}_{jk}^{\tau} - \Sigma_{jk}^0\right| \leq \frac{4\pi}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}} \ . \tag{9.17}$$

Proof: Let $0 < \alpha < 1$. According to (9.14), we have

$$\begin{aligned}\mathbb{P}_X(|\hat{\rho}_{jk} - \mathbb{E}(\hat{\rho}_{jk})| > t) &\leq \mathbb{P}_X(|U_{jk} - \mathbb{E}(U_{jk})| > \alpha t) \\ &\quad + \mathbb{P}_X\left(\frac{3}{n_{jk}+1}|\hat{\tau}_{jk} - \tau_{jk}| > (1-\alpha)t\right) \ . \end{aligned} \tag{9.18}$$

Since $-1 \leq \tau_{jk} \leq 1$, $|\hat{\tau}_{jk} - \tau_{jk}| \leq 2$, then

$$\begin{aligned}&\mathbb{P}_X\left(\frac{3}{n_{jk}+1}|\hat{\tau}_{jk} - \tau_{jk}| > (1-\alpha)t\right) \\ &\leq \mathbb{P}_X\left(\frac{6}{n_{jk}+1} > (1-\alpha)t\right) \ . \end{aligned} \tag{9.19}$$

Applying Hoeffding's bound for U-statistics, we have

$$\begin{aligned}&\mathbb{P}_X(|U_{jk}-\mathbb{E}(U_{jk})|>\alpha t)\\ &\leq \exp\left(-2\left\lfloor\frac{n_{jk}}{3}\right\rfloor\frac{\alpha^2t^2}{36}\right)=\exp\left(-\frac{n_{jk}\alpha^2t^2}{54}\right)\ .\end{aligned}\tag{9.20}$$

Combining (9.19) and (9.20) yields

$$\begin{aligned}\mathbb{P}_X\left(|\hat{\rho}_{jk}-\ \mathbb{E}(\hat{\rho}_{jk})|>t\right)\leq \exp\left(-\frac{n_{jk}\alpha^2t^2}{54}\right)\\ +\mathbb{P}_X\left(\frac{6}{n_{jk}+1}>(1-\alpha)t\right)\ .\end{aligned}\tag{9.21}$$

In particular, if $n_{jk}\geq\frac{6}{(1-\alpha)t}$, the second term on the RHS is 0. Since $\hat{\rho}_{jk}$ is a biased estimator, following [118], we use the following bias equation [224]:

$$\mathbb{E}\hat{\rho}_{jk}=\frac{6}{\pi(n_{jk}+1)}\left[\arcsin(\Sigma^0_{jk})+(n_{jk}-2)\arcsin(\frac{\Sigma^0_{jk}}{2})\right]\ .\tag{9.22}$$

Note we only use n_{jk} effective number of samples. Thus,

$$\Sigma^0_{jk}=2\sin\left(\frac{\pi}{2}\mathbb{E}\hat{\rho}_{jk}+a_{jk}\right)\ ,\tag{9.23}$$

where

$$a_{jk}=\frac{\pi\mathbb{E}\hat{\rho}_{jk}-2\arcsin(\Sigma^0_{jk})}{2(n_{jk}-2)}\ ,|a_{jk}|\leq\frac{\pi}{n_{jk}-2}\ .\tag{9.24}$$

If $n_{jk}\geq\frac{6\pi}{t}+2$, $|a_{jk}|\leq\frac{t}{6}$. Therefore, the analysis is simplified if $\inf_{jk}n_{jk}\geq c_0$ where

$$c_0\geq\max\left\{\frac{6}{(1-\alpha)t},\frac{6\pi}{t}+2\right\}\ .\tag{9.25}$$

Setting $\alpha=\frac{3\sqrt{6}}{8}, t=\frac{4\pi}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}}$, we have

$$\begin{aligned}\frac{6}{(1-\alpha)t}&\leq\frac{24\pi}{t}=6(1-\delta)\sqrt{\frac{1-\epsilon}{m+2}}\sqrt{\frac{n}{\log p}}\ ,\\ \frac{6\pi}{t}+2&=\frac{3(1-\delta)}{2}\sqrt{\frac{1-\epsilon}{m+2}}\sqrt{\frac{n}{\log p}}\ .\end{aligned}$$

Therefore, we choose

$$c_0=6(1-\delta)\sqrt{\frac{1-\epsilon}{m+2}}\sqrt{\frac{n}{\log p}}\ .\tag{9.26}$$

Define an event $Z = \{\inf_{jk} n_{jk} \geq c_0\}$, and let $\bar{Z}$ be the complement of the event. Further, the event of interest is $Y = \left\{ \sup_{j,k} \left| \hat{S}^{\tau}_{jk} - \Sigma^0_{jk} \right| \leq \frac{4\pi}{1-\delta} \sqrt{\frac{m+2}{1-\epsilon}} \sqrt{\frac{\log p}{n}} \right\}$. Then, the probability of the event of interest can be lower bounded as:

$$\begin{aligned} P(Y) &= P(Y|Z)P(Z) + P(Y|\bar{Z})P(\bar{Z}) \\ &\geq P(Y|Z)P(Z)\ . \end{aligned} \tag{9.27}$$

Next, we focus on getting lower bounds to both $P(Z)$ and $P(Y|Z)$.

Note $n_{jk} = \sum_{i=1}^{n} b_{ij} b_{ik}$ and $\mathbb{E}[n_{jk}] = (1-\delta)^2 n$, using Chernoff bounds,

$$\mathbb{P}_B\left(n_{jk} < (1-\epsilon)(1-\delta)^2 n\right) \leq \exp\left(-\epsilon^2 (1-\delta)^2 n/2\right)\ . \tag{9.28}$$

By the union bound,

$$\begin{aligned} &\mathbb{P}_B\left(\inf_{jk} n_{jk} < (1-\epsilon)(1-\delta)^2 n\right) \\ &\leq \exp\left(-\epsilon^2 (1-\delta)^2 n/2 + 2\log p\right)\ , \end{aligned} \tag{9.29}$$

which is equivalent to

$$\begin{aligned} &\mathbb{P}_B\left(\inf_{jk} n_{jk} \geq (1-\epsilon)(1-\delta)^2 n\right) \\ &\geq 1 - \exp\left(-\epsilon^2 (1-\delta)^2 n/2 + 2\log p\right)\ . \end{aligned} \tag{9.30}$$

If $(1-\epsilon)(1-\delta)^2 n \geq c_0$, i.e.,

$$n \geq \frac{36}{(m+2)(1-\epsilon)(1-\delta)^2 \log p}\ , \tag{9.31}$$

then

$$\mathbb{P}_B\left(\inf_{jk} n_{jk} \geq c_0\right) \geq 1 - \exp\left(-\epsilon^2 (1-\delta)^2 n/2 + 2\log p\right)\ , \tag{9.32}$$

which gives a lower bound to $P(Z)$ as desired. Now, conditioned on Z, i.e., $\inf_{jk} n_{jk} \geq c_0$, we have $|a_{jk}| \leq \frac{t}{6}$, and $\mathbb{P}_X\left(\frac{6}{n_{jk}+1} > (1-\alpha)t \,\middle|\, Z\right) = 0$. Assuming n satisfies (9.31) and using (9.21), (9.23), we have

$$\mathbb{P}_X\left(|\hat{S}^{\rho}_{jk} - \Sigma^0_{jk}| > t \,\middle|\, Z\right)$$

$$
\begin{aligned}
&= \mathbb{P}_X\left(\left|2\sin\left(\frac{\pi}{6}\hat{\rho}_{jk}\right) - 2\sin(\frac{\pi}{6}\mathbb{E}\hat{\rho}_{jk} + a_{jk})\right| > t \,\middle|\, Z\right) \\
&\le \mathbb{P}_X\left(\left|\frac{\pi}{3}\hat{\rho}_{jk} - \frac{\pi}{3}\mathbb{E}\hat{\rho}_{jk} - 2a_{jk}\right| > t \,\middle|\, Z\right) \\
&= \mathbb{P}_X\left(\left|\hat{\rho}_{jk} - \mathbb{E}\hat{\rho}_{jk} - \frac{6}{\pi}a_{jk}\right| > \frac{3t}{\pi} \,\middle|\, Z\right) \\
&\le \mathbb{P}_X\left(|\hat{\rho}_{jk} - \mathbb{E}\hat{\rho}_{jk}| > \frac{3t}{\pi} - \left|\frac{6}{\pi}a_{jk}\right| \,\middle|\, Z\right) \\
&\le \mathbb{P}_X\left(|\hat{\rho}_{jk} - \mathbb{E}\hat{\rho}_{jk}| > \frac{2t}{\pi} \,\middle|\, Z\right) \\
&\le \exp\left(-\frac{2n_{jk}\alpha^2 t^2}{27\pi^2}\right) , \qquad (9.33)
\end{aligned}
$$

where the conditioning on Z, i.e., $\{\inf_{j,k} n_{jk} \ge c_0\}$, has been dropped in the last inequality yielding an upper bound. Setting $\alpha = \frac{3\sqrt{6}}{8}, t = \frac{4\pi}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}}$, by the union bound, we have

$$
\begin{aligned}
&\mathbb{P}_X\left(\sup_{jk}|\hat{S}^{\rho}_{jk} - \Sigma^0_{jk}| > t \,\middle|\, Z\right) \\
&\le \sum_{j,k} \exp\left(-\frac{n_{jk}}{(1-\delta)^2(1-\epsilon)n}(m+2)\log p\right) , \qquad (9.34)
\end{aligned}
$$

which is the same as (9.12). Using Lemma 34, we then have $P(Y|Z) \ge \left(1 - \frac{1}{p^m}\right)$. The result of the theorem then follows from (9.27) and (9.30). ■

9.3.3 Plug-in CLIME Estimator

Since $\hat{S}$ ($\hat{S}^{\tau}$ or $\hat{S}^{\rho}$) satisfies (9.10) or (9.17) with high probability, choosing $\lambda_n \ge \frac{\pi\|\Omega^0\|_{L_1}}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}}$ or $\lambda_n \ge \frac{4\pi\|\Omega^0\|_{L_1}}{1-\delta}\sqrt{\frac{m+2}{1-\epsilon}}\sqrt{\frac{\log p}{n}}$ ensures that the conditions for consistency of the CLIME estimate $\hat{\Omega}$ are satisfied. The CLIME estimator considers the following family of precision matrices $\mathcal{U} = \mathcal{U}(M, q, s_0(p)) = \left\{\Omega : \Omega \succ 0, \|\Omega\|_{L_1} \le M, \max_{1\le i\le p}\sum_{j=1}^{p}|\omega_{ij}|^q \le s_0(p)\right\}$, for $0 \le q < 1$. Then, the CLIME estimator has the following guarantees:

Theorem 38 *Let $\Omega_0 \in \mathcal{U}(M, q, s_0(p))$. If $\lambda_n \ge \|\Omega_0\|_{L_1}\max_{ij}|\hat{\sigma}_{n,ij} - \sigma_{0,ij}|$, then we have*

$$|\hat{\Omega}_n - \Omega_0|_\infty \le 4\|\Omega_0\|_{L_1}\lambda_n , \qquad (9.35)$$

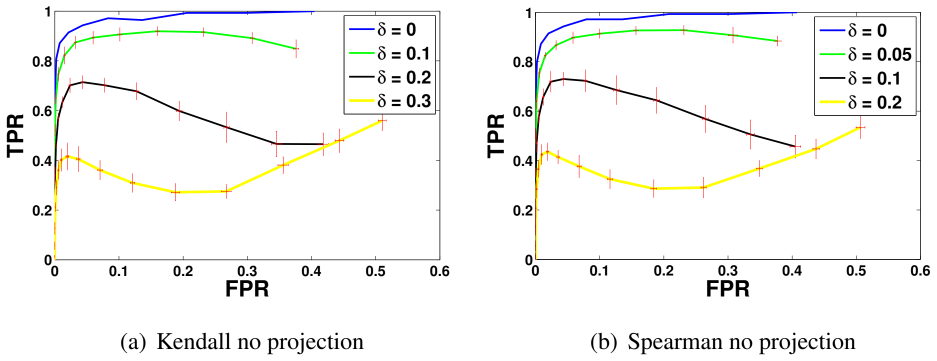
$$\|\hat{\Omega}_n - \Omega_0\|_2 \le Cs_0(p)(4\|\Omega_0\|_{L_1})^{1-q}\lambda_n^{1-q} , \qquad (9.36)$$

$$\frac{1}{p}\|\hat{\Omega}_n - \Omega_0\|_F^2 \leq Cs_0(p)(4\|\Omega_0\|_{L_1})^{2-q}\lambda_n^{2-q}\,, \tag{9.37}$$

where $C \leq 2(1 + 2^{1-q} + 3^{1-q})$ *is a constant.*

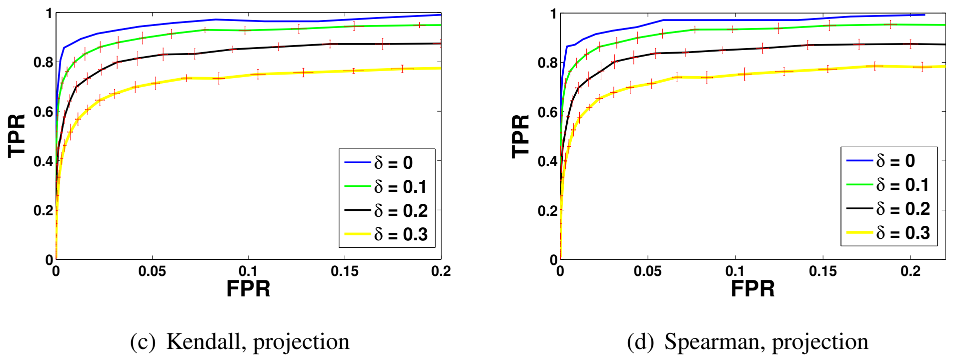
Note that deterministic bounds in Theorem 38 for precision estimation relies on $|\hat{\Sigma}_n - \Sigma_0|_\infty = \max_{i,j}|\hat{\sigma}_{n,ij} - \sigma_{0,ij}|$.

9.4 Experimental Results



(a) Kendall no projection

(b) Spearman no projection



(c) Kendall, projection

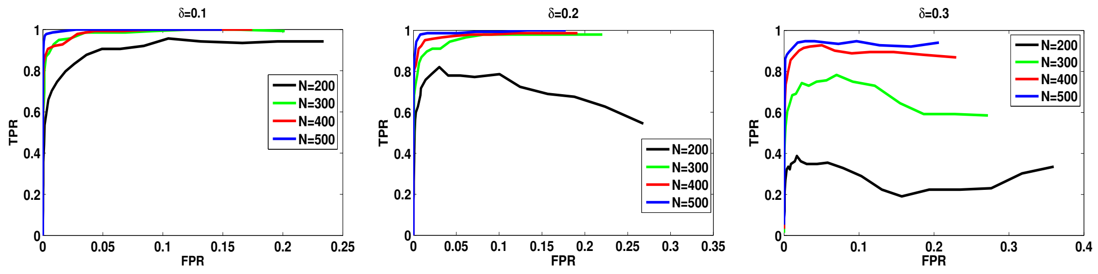
(d) Spearman, projection

Figure 9.1: (a,b) ROC curves without projection ($\hat{S}$ need not be positive semi-definite), (c,d) ROC curves with projection ($\hat{S}$ is positive semi-definite) with $n = 200$ and under different missing probabilities ($\delta = 0.1 - 0.3$). By increasing number of observed data (smaller δ), the ROC curve approaches the ROC curve of no-missing data ($\delta = 0$).

We present experimental results of DoPinG on both synthetic datasets and real datasets to illustrate model performance. The first set of experiments on synthetic data illustrate the effect of sample size and percentage of missing data on model performance. Then we compare DoPinG with mGlasso on both synthetic data and climate dataset.

9.4.1 Synthetic Data

To generate synthetic data, we use the procedure described in [118]. First, a d-dimensional sparse graph $G = (V, E)$ is generated as follows: Let $V = \{1, ..., p\}$ correspond to variables $X = (X_1, ..., X_d)$. We associate each index j with a bivariate point $Y_j = (Y_j^{(1)}, Y_j^{(2)}) \in [0, 1]^2$ where each $Y_j^{(k)} \sim \text{Unif}[0, 1],\ k = 1, 2,\ j \in \{1, \cdots, d\}$. An edge is associated between vertices (i, j) with probability of $P\left((i, j) \in E\right) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\|y_i - y_j\|^2}{0.25}\right)$ where $y_j = (y_j^{(1)}, y_j^{(2)})$ is the observation of Y_j and $\| \, . \, \|$ denotes the Euclidean distance. The maximum degree of the graph is limited to 4. Thereafter, n samples are drawn from $NPN_d(f^0, \Sigma^0)$ where f^0 is the Gaussian CDF Transformation with mean 0.05 and standard deviation 0.4. Here, we choose $n = 200$, $p = 100$, and $\delta \in \{0.1, 0.2, 0.3\}$. The final results shown below are averages over 10 experimental runs for both Kendall's tau and Spearman's rho. The ROC curve is generated by varying the tuning parameter λ in the CLIME and calculating the corresponding False Positive Rate (FPR) and True Positive Rate (TPR) [118].



(a) $\delta = 10\%$ (b) $\delta = 20\%$ (c) $\delta = 30\%$

Figure 9.2: ROC curve with $\delta = 0.1, 0.2, 0.3$, $p = 100$, and different number of samples (n). For a fixed value of δ, with increasing number of samples, the higher TP rates is obtained.

First, we directly run Algorithm 7 using $\hat{S}$ ($\hat{S}^{\tau}$ or $\hat{S}^{\rho}$) estimated using Kendall's tau and Spearman's rho. The ROC curve with different probabilities of missing values is plotted in Figure 9.1. We observe that the performance of Kendall's tau and Spearman's rho is almost the same for the same percentage of missing values. Note that the tuning parameter λ controls the sparsity of the estimated graph, i.e., a small value of λ provides a dense graph. When λ is large enough the predicted edges are all among the correct edges leading to a zero FPR. By decreasing λ, false edges that are not in the original graph are added, i.e., increasing FPR and saturating TPR. It shows that the estimator is conservative in adding edges. Figure 9.1 also illustrates that increasing number of missing values (increasing δ) deteriorates model performance, while increasing variance of estimate.

As mentioned in section 9.2.3, the estimated correlation matrix $\hat{S}$ may be not positive semi-definite. Therefore, we project $\hat{S}$ into the positive semi-definite (PSD) cone, and execute Algorithm 7 using the PSD matrix. Figures 9.1 (c,d) plot the ROC curve with projection for Kendall's tau and Spearman's rho respectively. For small δ, e.g. $\delta = 0.1$, to some degree, the performances with and without projection are similar. However, when more values are missing, PSD projection greatly improves performance. Increasing percentage of missing values lead to more and larger negative eigenvalues in $\hat{S}$, and performance worsens for higher δ. Note that our analysis shows that the *effective* sample size is $(1-\delta)^2 n$, and decrease of the recovery rate (TPR) with decreasing *effective* sample size is in accordance with our analysis. In other words, for a fixed n the *effective* sample size is smaller for a larger value of δ and therefore, DoPinG has a worse performance with larger value of δ.

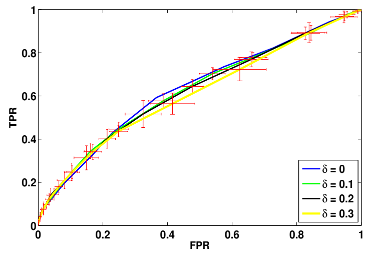
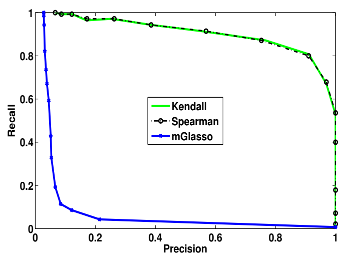
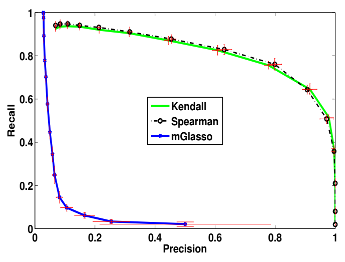


Figure 9.3: ROC curve of mGlasso with $n = 200$ and different missing probabilities. mGlasso has a worse performance on non-Gaussian data compared to DoPinG (Figure 9.1).

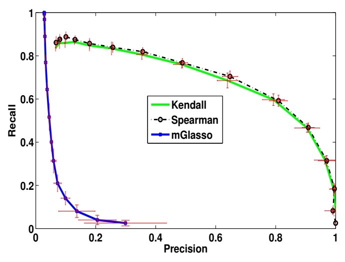
Figure 9.2 shows the effect of sample size n with different value of δ on the performance without projection. Under higher percentage of missing values (Figure 9.2(c)), the performance of the method suffers much more with low sample size, compared to data with lower percentage of missing entries (Figure 9.2(a)). In particular, with a sample size $n = 200$ and 30% of missing data, the *effective* sample size is ~ 100 while with 10% of missing data, the *effective* sample size is ~ 160. As a result, to achieve similar recovery rates (TPR,FPR), higher sample size is needed when more percentage of the data is missing.



(a) $\delta = 0$

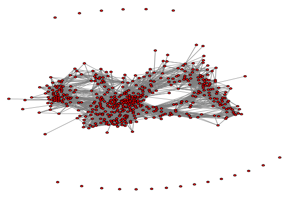


(b) $\delta = 10\%$

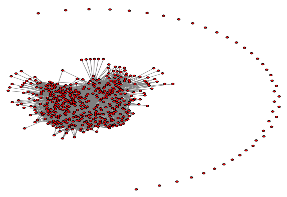


(c) $\delta = 20\%$

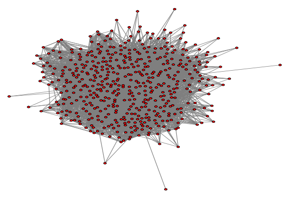
Figure 9.4: Precision and Recall Curve with different δ. DoPinG is significantly better than mGlasso for non-Gaussian data.



(a) DoPinG (12240 edges)



(b) mGlasso (8778 edges)



(c) mGlasso (11860 edges)

Figure 9.5: The graph discovered by DoPinG and mGlasso.

We compare DoPinG with mGlasso [100] on the synthetic data. The ROC curve of mGlasso is plotted in Figure 9.3. Since mGlasso is designed primarily for Gaussian data, Figure 9.3 clearly illustrates that mGlasso is not suitable for non-Gaussian data. We also plot the precision and recall curve with different probabilities of missing values ($\delta = 0, 0.1, 0.2$) in Figure 9.4. The performance of DoPinG is significantly better than mGlasso.

9.4.2 Climate Data

We compare DoPinG (Spearman's rho) and mGlasso on Climate data. The climate dataset that we use is obtained from the CMIP5 archive, where we use the temperature predicted over land locations by a climate model. We reduce the resolution of the data, since we use it only for illustrative purposes, so that the data contains 500 locations (dimensionality), and yearly averaged samples over 100 years (sample size =100). We randomly remove $\delta = 20\%$ of the entries. We try different λ and report the results which have similar number of edges. In particular, we

Table 9.1: Edges dicovered by DoPinG and mGlasso on Climate Data. $>$ denotes the number of edges in DoPinG graph but not in mGlasso graph. $<$ is on the contrary.

Edge No.		Edge Diff	
DoPinG	mGlasso	$>$	$<$
12240	8778	7942	4480
12240	11860	7534	7154

pick the graph with 12740 edges for DoPinG ($\lambda = 0.02$) as illustrated in Figure 9.5(a). We pick two graphs for mGlasso. One has 8778 edges ($\lambda = 0.001$) and the other has 11860 edges ($\lambda = 0.002$), as shown in Figure 9.5(b) and 9.5(c) respectively. It seems that DoPinG discovers some interesting sparsity patterns while mGlasso graphs are messy. In Table 1, we present the difference between DoPinG graph and mGlasso graph. With similar total number of edges, DoPinG graph shows more structure than mGlasso graph. We plan to further investigate this behavior in future work.

Chapter 10

Online ℓ_1-Dictionary Learning with Application to Novel Document Detection

10.1 Introduction

The high volume and velocity of social media, such as blogs and Twitter, have propelled them to the forefront as sources of breaking news. On Twitter, it is possible to find the latest updates on diverse topics, from natural disasters to celebrity deaths; and identifying such emerging topics has many practical applications, such as in marketing, disease control, and national security [137]. The key challenge in automatic detection of breaking news, is being able to detect novel documents in a stream of text; where a document is considered novel if it is "unlike" documents seen in the past. Recently, this has been made possible by *dictionary learning*, which has emerged as a powerful data representation framework. In dictionary learning each data point $\mathbf{y}$ is represented as a sparse linear combination $A\mathbf{x}$ of dictionary atoms, where A is the dictionary and $\mathbf{x}$ is a sparse vector [2, 129]. A dictionary learning approach can be easily converted into a novel document detection method: let A be a dictionary representing all documents till time $t-1$, for a new data document $\mathbf{y}$ arriving at time t, if one does not find a sparse combination $\mathbf{x}$ of the dictionary atoms, and the best reconstruction $A\mathbf{x}$ yields a large loss, then $\mathbf{y}$ clearly is not well represented by the dictionary A, and is hence novel compared to documents in the past. At the end of timestep t, the dictionary is updated to represent all the documents till time t.

Kasiviswanathan *et al.* [97] presented such a (batch) dictionary learning approach for detecting novel documents/topics. They used an ℓ_1-penalty on the reconstruction error (instead of squared loss commonly used in the dictionary learning literature) as the ℓ_1-penalty has been found to be more effective for text analysis (see Section 10.3). They also showed this approach outperforms other techniques, such as a nearest-neighbor approach popular in the related area of *First Story Detection* [155]. We build upon this work, by proposing an efficient algorithm for online dictionary learning with ℓ_1-penalty. Our online dictionary learning algorithm is based on the *online alternating directions* method which was recently proposed by Wang and Banerjee [200] to solve online composite optimization problems with additional linear equality constraints. Traditional online convex optimization methods such as [225, 75, 53, 52, 210] require explicit computation of the subgradient making them computationally expensive to be applied in our high volume text setting, whereas in our algorithm the subgradients are computed implicitly. The algorithm has simple closed form updates for all steps yielding a fast and scalable algorithm for updating the dictionary. Under suitable assumptions (to cope with the non-convexity of the dictionary learning problem), we establish an $O(\sqrt{T})$ regret bound for the objective, matching the regret bounds of existing methods [225, 53, 52, 210]. Using this online algorithm for ℓ_1-dictionary learning, we obtain an online algorithm for novel document detection, which we empirically validate on traditional news-streams as well as streaming data from Twitter. Experimental results show a substantial speedup over the batch ℓ_1-dictionary learning based approach of Kasiviswanathan *et al.* [97], without a loss of performance in detecting novel documents.

Related Work. Online convex optimization is an area of active research and for a detailed survey on the literature we refer the reader to [177]. Online dictionary learning was recently introduced by Mairal *et al.* [129] who showed that it provides a scalable approach for handling large dynamic datasets. They considered an ℓ_2-penalty and showed that their online algorithm converges to the minimum objective value in the stochastic case (i.e., with distributional assumptions on the data). However, the ideas proposed in [129] do not translate to the ℓ_1-penalty. The problem of novel document/topics detection was also addressed by a recent work of Saha *et al.* [170], where they proposed a non-negative matrix factorization based approach for capturing evolving and novel topics. However, their algorithm operates over a sliding time window (does not have online regret guarantees) and works only for ℓ_2-penalty.

10.2 Preliminaries

Notation. Vectors are always column vectors and are denoted by boldface letters. For a matrix Z its norm, $\|Z\|_1 = \sum_{i,j} |z_{ij}|$ and $\|Z\|_F^2 = \sum_{ij} z_{ij}^2$. For arbitrary real matrices the standard inner product is defined as $\langle Y, Z \rangle = \mathrm{Tr}(Y^\top Z)$. We use $\Psi_{\max}(Z)$ to denote the largest eigenvalue of $Z^\top Z$. For a scalar $r \in \mathbb{R}$, let $\mathrm{sign}(r) = 1$ if $r > 0$, -1 if $r < 0$, and 0 if $r = 0$. Define $\mathrm{soft}(r, T) = \mathrm{sign}(r) \cdot \max\{|r| - T, 0\}$. The operators sign and soft are extended to a matrix by applying it to every entry in the matrix. $\mathbf{0}_{m \times n}$ denotes a matrix of all zeros of size $m \times n$ and the subscript is omitted when the dimension of the represented matrix is clear from the context.

Dictionary Learning Background. *Dictionary learning* is the problem of estimating a collection of basis vectors over which a given data collection can be accurately reconstructed, often with sparse encodings. It falls into a general category of techniques known as *matrix factorization*. Classic dictionary learning techniques for sparse representation (see [2, 151, 129] and references therein) consider a finite training set of signals $P = [\mathbf{p}_1, \dots, \mathbf{p}_n] \in \mathbb{R}^{m \times n}$ and optimize the empirical cost function which is defined as $f(A) = \sum_{i=1}^n l(\mathbf{p}_i, A)$, where $l(\cdot, \cdot)$ is a loss function such that $l(\mathbf{p}_i, A)$ should be small if A is "good" at representing the signal $\mathbf{p}_i$ in a sparse fashion. Here, $A \in \mathbb{R}^{m \times k}$ is referred to as the *dictionary*. In this chapter, we use a ℓ_1-loss function with an ℓ_1-regularization term, and our

$$l(\mathbf{p}_i, A) = \min_{\mathbf{x}} \ \|\mathbf{p}_i - A\mathbf{x}\|_1 + \lambda \|\mathbf{x}\|_1, \text{ where } \lambda \text{ is the regularization parameter.}$$

We define the problem of dictionary learning as that of minimizing the empirical cost $f(A)$. In other words, the dictionary learning is the following optimization problem

$$\min_A f(A) = f(A, X) \overset{\text{def}}{=} \min_{A,X} \sum_{i=1}^n l(\mathbf{p}_i, A) = \min_{A,X} \ \|P - AX\|_1 + \lambda \|X\|_1.$$

For maintaining interpretability of the results, we would additionally require that the A and X matrices be non-negative. To prevent A from being arbitrarily large (which would lead to arbitrarily small values of X), we add a scaling constant on A as follows. Let $\mathcal{A}$ be the convex set of matrices defined as

$$\mathcal{A} = \{A \in \mathbb{R}^{m \times k} \ : \ A \geq \mathbf{0}_{m \times k} \ \ \forall j = 1, \dots, k \ , \|A_j\|_1 \leq 1\}, \text{ where } A_j \text{ is the } j\text{th column in } A.$$

We use $\Pi_{\mathcal{A}}$ to denote the Euclidean projection onto the nearest point in the convex set $\mathcal{A}$. The resulting optimization problem can be written as

$$\min_{A \in \mathcal{A}, X \geq \mathbf{0}} \ \|P - AX\|_1 + \lambda \|X\|_1 \tag{10.1}$$

The optimization problem (10.1) is in general non-convex. But if one of the variables, either A or X is known, the objective function with respect to the other variable becomes a convex function (in fact, can be transformed into a linear program).

10.3 Novel Document Detection Using Dictionary Learning

In this section, we describe the problem of novel document detection and explain how dictionary learning could be used to tackle this problem. Our problem setup is similar to [97].

Novel Document Detection Task. We assume documents arrive in streams. Let $\{P_t : P_t \in \mathbb{R}^{m_t \times n_t}, t = 1, 2, 3, \dots\}$ denote a sequence of streaming matrices where each column of P_t represents a document arriving at time t. Here, P_t represents the term-document matrix observed at time t. Each document is represented is some conventional vector space model such as TF-IDF [130]. The t could be at any granularity, e.g., it could be the day that the document arrives. We use n_t to represent the number of documents arriving at time t. We normalize P_t such that each column (document) in P_t has a unit ℓ_1-norm. For simplicity in exposition, we will assume that $m_t = m$ for all t.[1] We use the notation $P_{[t]}$ to denote the term-document matrix obtained by vertically concatenating the matrices $P_1, \dots, P_t$, i.e., $P_{[t]} = [P_1|P_2|\dots|P_t]$. Let N_t be the number of documents arriving at time $\leq t$, then $P_{[t]} \in \mathbb{R}^{m \times N_t}$. Under this setup, the *goal* of novel document detection is to identify documents in P_t that are "dissimilar" to the documents in $P_{[t-1]}$.

Sparse Coding to Detect Novel Documents. Let $A_t \in \mathbb{R}^{m \times k}$ represent the dictionary matrix after time $t-1$; where dictionary A_t is a good basis to represent of all the documents in $P_{[t-1]}$. The exact construction of the dictionary is described later. Now, consider a document $\mathbf{y} \in \mathbb{R}^m$ appearing at time t. We say that it admits a *sparse* representation over A_t, if $\mathbf{y}$ could be "well" approximated as a linear combination of few columns from A_t. Modeling a vector with such a sparse decomposition is known as *sparse coding*. In most practical situations it may not be possible to represent $\mathbf{y}$ as $A_t\mathbf{x}$, e.g., if $\mathbf{y}$ has new words which are absent in A_t. In such cases, one could represent $\mathbf{y} = A_t\mathbf{x} + \mathbf{e}$ where $\mathbf{e}$ is an unknown noise vector. We consider the

[1] As new documents come in and new terms are identified, we expand the vocabulary and zero-pad the previous matrices so that at the current time t, all previous and current documents have a representation over the same vocabulary space.

following sparse coding formulation

$$l(\mathbf{y}, A_t) = \min_{\mathbf{x} \geq \mathbf{0}} \ \|\mathbf{y} - A_t\mathbf{x}\|_1 + \lambda\|\mathbf{x}\|_1. \tag{10.2}$$

The formulation (10.2) naturally takes into account both the reconstruction error (with the $\|\mathbf{y} - A_t\mathbf{x}\|_1$ term) and the complexity of the sparse decomposition (with the $\|\mathbf{x}\|_1$ term). The reconstruction error measures the quality of the approximation while the complexity is measured by the ℓ_1-norm of the optimal $\mathbf{x}$. It is quite easy to transform (10.2) into a linear program. Hence, it can be solved using a variety of methods. In our experiments, we use the alternating directions method of multipliers (ADMM) [19] to solve (10.2). ADMM has recently gathered significant attention in the machine learning community due to its wide applicability to a range of learning problems with complex objective functions [19].

We can use sparse coding to detect novel documents as follows. For each document $\mathbf{y}$ arriving at time t, we do the following. First, we solve (10.2) to check whether $\mathbf{y}$ could be well approximated as a sparse linear combination of the atoms of A_t. If the objective value $l(\mathbf{y}, A_t)$ is "big" then we mark the document as *novel*, otherwise we mark the document as *non-novel*. Since, we have normalized all documents in P_t to unit ℓ_1-length, the objective values are in the same scale.

Choice of the Error Function. A very common choice of reconstruction error is the ℓ_2-penalty. In fact, in the presence of isotopic Gaussian noise the ℓ_2-penalty on $\mathbf{e} = \mathbf{y} - A_t\mathbf{x}$ gives the maximum likelihood estimate of $\mathbf{x}$ [209, 213]. However, for text documents, the noise vector $\mathbf{e}$ rarely satisfies the Gaussian assumption, as some of its coefficients contain large, impulsive values. For example, in fields such as politics and sports, a certain term may become suddenly dominant in a discussion [97]. In such cases imposing an ℓ_1-penalty on the error is a better choice than imposing an ℓ_2-penalty (e.g., recent research [209, 214, 208] have successfully shown the superiority of ℓ_1 over ℓ_2 penalty for a different but related application domain of face recognition). We empirically validate the superiority of using the ℓ_1-penalty for novel document detection in Section 10.5.

Size of the Dictionary. Ideally, in our application setting, changing the size of the dictionary (k) dynamically with t would lead to a more efficient and effective sparse coding. However, in our theoretical analysis, we make the simplifying assumption that k is a constant independent of t. In our experiments, we allow for small increases in the size of the dictionary over time when required. The problem of designing an adaptive dictionary whose size automatically increase or decrease over time is an interesting open problem.

Batch Algorithm for Novel Document Detection. We now describe a simple batch algorithm (slightly modified from [97]) for detecting novel documents. The Algorithm 10.3 alternates between a novel document detection and a batch dictionary learning step.

```
1: Input: $P_{[t-1]} \in \mathbb{R}^{m \times N_{t-1}}$, $P_t = [\mathbf{p}_1, \dots, \mathbf{p}_{n_t}] \in \mathbb{R}^{m \times n_t}$, $A_t \in \mathbb{R}^{m \times k}$, $\lambda \geq 0, \zeta \geq 0$
2: Novel Document Detection Step:
3: for $j = 1$ to $n_t$ do
4:     Solve: $\mathbf{x}_j = \text{argmin}_{\mathbf{x} \geq 0} \|\mathbf{p}_j - A_t\mathbf{x}\|_1 + \lambda\|\mathbf{x}\|_1$
5:     if $\|\mathbf{p}_j - A_t\mathbf{x}_j\|_1 + \lambda\|\mathbf{x}_j\|_1 > \zeta$
6:         Mark $\mathbf{p}_j$ as novel
7: Batch Dictionary Learning Step:
8: Set $P_{[t]} \leftarrow [P_{[t-1]} \,|\, \mathbf{p}_1, \dots, \mathbf{p}_{n_t}]$
9: Solve: $[A_{t+1}, X_{[t]}] = \text{argmin}_{A \in \mathcal{A}, X \geq \mathbf{0}} \|P_{[t]} - AX\|_1 + \lambda\|X\|_1$
```

Batch Dictionary Learning. We now describe the batch dictionary learning step from Algorithm 10.3. At time t, the dictionary learning step is[2]

$$[A_{t+1}, X_{[t]}] = \text{argmin}_{A \in \mathcal{A}, X \geq \mathbf{0}} \|P_{[t]} - AX\|_1 + \lambda\|X\|_1. \tag{10.3}$$

Even though conceptually simple, Algorithm 10.3 is computationally inefficient. The bottleneck comes in the dictionary learning step. As t increases, so does the size of $P_{[t]}$, so solving (10.3) becomes prohibitive even with efficient optimization techniques. To achieve computational efficiency, in [97], the authors solved an approximation of (10.3) where in the dictionary learning step they only update the A's and not the X's.[3] This leads to faster running times, but because of the approximation, the quality of the dictionary degrades over time and the performance of the algorithm decreases. In this chapter, we propose an online learning algorithm for (10.3) and show that this online algorithm is both computationally efficient and generates good quality dictionaries under reasonable assumptions.

[2] In our algorithms, it is quite straightforward to replace the condition $A \in \mathcal{A}$ by some other condition $A \in \mathcal{C}$, where $\mathcal{C}$ is some closed non-empty convex set.

[3] In particular, define (recursively) $\widetilde{X}_{[t]} = [\widetilde{X}_{[t-1]} \,|\, \mathbf{x}_1, \dots, \mathbf{x}_{n_t}]$ where $\mathbf{x}_j$'s are coming from the novel document detection step at time t. In [97], the dictionary learning step is $A_{t+1} = \text{argmin}_{A \in \mathcal{A}} \|P_{[t]} - A\widetilde{X}_{[t]}\|_1$.

10.4 Online ℓ_1-Dictionary Learning

In this section, we introduce the online ℓ_1-dictionary learning problem and propose an efficient algorithm for it. The standard goal of online learning is to design algorithms whose regret is sublinear in time T, since this implies that "on the average" the algorithm performs as well as the best fixed strategy in hindsight [177]. Now consider the ℓ_1-dictionary learning problem defined in (10.3). Since this problem is non-convex, it may not be possible to design polynomial running time offline (batch) algorithms that solve it without making any assumptions on either the dictionary (A) or the sparse code (X). This also means that it may not be possible to design a polynomial time online algorithm with sublinear regret without making any assumptions on either A or X because a polynomial time online algorithm with sublinear regret would imply would imply a polynomial time offline algorithm for solving (10.1). Therefore, we focus on obtaining regret bounds for the dictionary update, assuming that the at each timestep the sparse codes given to the batch and online algorithms are "close". This motivates the following problem.

Definition 1 (Online ℓ_1-Dictionary Learning Problem) At time $t = 0, 1, \ldots,$ the online algorithm picks $\hat{A}_{t+1} \in \mathcal{A}$. Then, the nature (adversary) reveals $(P_{t+1}, \hat{X}_{t+1})$ with $P_{t+1} \in \mathbb{R}^{m \times n}$ and $\hat{X}_{t+1} \in \mathbb{R}^{k \times n}$. The problem is to pick the $\hat{A}_{t+1}$ sequence such that the following regret function is minimized[4]

$$R(T) = \sum_{t=1}^{T} \|P_t - \hat{A}_t \hat{X}_t\|_1 - \min_{A \in \mathcal{A}} \sum_{t=1}^{T} \|P_t - AX_t\|_1 \ ,$$

where $\hat{X}_t = X_t + E_t$ and E_t is an error matrix dependent on t.

The regret defined above admits the discrepancy between the sparse coding matrices supplied to the batch and online algorithms through the error matrix. The reason for this generality is because in our application setting, the sparse coding matrices used for updating the dictionaries of the batch and online algorithms could be different. We will later establish the conditions on E_t's under which we can achieve sublinear regret.

[4] For ease of presentation and analysis, we will assume that m and n don't vary with time. One could allow for changing m and n by carefully adjusting the size of the matrices by zero-padding.

10.4.1 Online ℓ_1-Dictionary Algorithm

In this section, we design an algorithm for the online ℓ_1-dictionary learning problem, which we call Online Inexact ADMM (OIADMM) and bound its regret. Firstly note that because of the non-smooth ℓ_1-norms involved it is computationally expensive to apply standard online learning algorithms like online gradient descent [225, 75], COMID [52], FOBOS [53], and RDA [210], as they require computing a costly subgradient at every iteration. The subgradient of $\|P - AX\|_1$ at $A = \bar{A}$ is $\text{sign}(\bar{A}X - P) \cdot X^\top$.

Our algorithm for online ℓ_1-dictionary learning is based on the online alternating direction method which was recently proposed by Wang *et al.* [200]. Our algorithm first performs a simple variable substitution by introducing an equality constraint. The update for each of the resulting variable has a closed-form solution without the need of estimating the subgradients explicitly.

Algorithm 8 : OIADMM

```
1: **Input:** $P_t \in \mathbb{R}^{m\times n}$, $\hat{A}_t \in \mathbb{R}^{m\times k}$, $\Delta_t \in \mathbb{R}^{m\times n}$, $\hat{X}_t \in \mathbb{R}^{k\times n}$, $\beta_t \geq 0$, $\tau_t \geq 0$
2: $\widetilde{\Gamma}_t \longleftarrow P_t - \hat{A}_t\hat{X}_t$
3: $\Gamma_{t+1} = \text{argmin}_\Gamma\ \|\Gamma\|_1 + \langle \Delta_t, \widetilde{\Gamma}_t - \Gamma\rangle + (\beta_t/2)\|\widetilde{\Gamma}_t - \Gamma\|_F^2$
4:                                    $(\Rightarrow \Gamma_{t+1} = \text{soft}(\widetilde{\Gamma}_t + \Delta_t/\beta_t, 1/\beta_t))$
5: $G_{t+1} \longleftarrow -(\Delta_t/\beta_t + \widetilde{\Gamma}_t - \Gamma_{t+1})\hat{X}_t^\top$
6: $\hat{A}_{t+1} = \text{argmin}_{A\in\mathcal{A}}\ \beta_t(\langle G_{t+1}, A - \hat{A}_t\rangle + (1/2\tau_t)\|A - \hat{A}_t\|_F^2\ )$
7:                                    $(\Rightarrow \hat{A}_{t+1} = \Pi_{\mathcal{A}}(\max\{0, \hat{A}_t - \tau_t G_{t+1}\}))$
8: $\Delta_{t+1} = \Delta_t + \beta_t(P_t - \hat{A}_{t+1}\hat{X}_t - \Gamma_{t+1})$
9: Return $\hat{A}_{t+1}$ and $\Delta_{t+1}$
```

The Algorithm 8 is simple. Consider the following minimization problem at time t

$$\min_{A\in\mathcal{A}} \|P_t - A\hat{X}_t\|_1.$$

We can rewrite this above minimization problem as:

$$\min_{A\in\mathcal{A},\Gamma} \|\Gamma\|_1 \quad \text{such that} \quad P_t - A\hat{X}_t = \Gamma. \tag{10.4}$$

The augmented Lagrangian of (10.4) is:

$$\mathcal{L}(A, \Gamma, \Delta) = \|\Gamma\|_1 + \langle \Delta, P_t - A\hat{X}_t - \Gamma\rangle + \frac{\beta_t}{2}\left\|P_t - A\hat{X}_t - \Gamma\right\|_F^2, \tag{10.5}$$

for $A \in \mathcal{A}$, $\Gamma \in \mathbb{R}^{m \times n}$, $\Delta \in \mathbb{R}^{m \times n}$, and $\beta_t > 0$. Here, Δ is a multiplier and β_t a penalty parameter.

OIADMM is summarized in Algorithm 8. The algorithm generates a sequence of iterates $\{\Gamma_t, \hat{A}_t, \Delta_t\}_{t=1}^{\infty}$. At each time t, instead of solving (10.4) completely, it only runs one step ADMM update of the variables $(\Gamma_t, \hat{A}_t, \Delta_t)$. Let $\widetilde{\Gamma}_t = P_t - \hat{A}_t \hat{X}_t$. The update steps are as follows.

1. First for a fixed $A = \hat{A}_t$ and Δ_t, Γ that minimizes (10.5) could be obtained by solving

$$\operatorname{argmin}_{\Gamma} \ \|\Gamma\|_1 + \langle \Delta_t, \widetilde{\Gamma}_t - \Gamma \rangle + (\beta_t/2)\|\widetilde{\Gamma}_t - \Gamma\|_F^2.$$

The Γ that minimizes this optimization problem is set as Γ_{t+1}.

2. Using $\Gamma = \Gamma_{t+1}$ and Δ_t, a simple manipulation shows that we can obtain the A that minimizes (10.5) by solving

$$\min_{A \in \mathcal{A}} \frac{\beta_t}{2} \left\| P_t - A\hat{X}_t - \Gamma_{t+1} + \frac{\Delta_t}{\beta_t} \right\|_F^2. \tag{10.6}$$

Instead of solving (10.6) exactly, we approximate it by

$$\min_{A \in \mathcal{A}} \beta_t(\langle G_{t+1}, A - \hat{A}_t \rangle + 1/(2\tau_t)\|A - \hat{A}_t\|_F^2),$$

where $\tau_t > 0$ is a proximal parameter and G_{t+1} is the gradient of $\|P_t - A\hat{X}_t - \Gamma_{t+1} + \Delta_t/\beta_t\|_F^2$ at $A = \hat{A}_t$. The above approach belongs to the class of proximal gradient methods in optimization [192, 214]. The A that minimizes this optimization problem is set as $\hat{A}_{t+1}$.

3. Update Δ as $\Delta_{t+1} = \Delta_t + \beta_t(P_t - \hat{A}_{t+1}\hat{X}_t - \Gamma_{t+1})$.

Equality Constraint Violation. OIADMM could temporary violate the equality constraint in (10.4), but satisfies the constraint on average in the long run. More formally, at each time t it could happen that $\hat{A}_{t+1}$ and Γ_{t+1} produced by OIADMM is such that $P_t - \hat{A}_{t+1}\hat{X}_t \neq \Gamma_{t+1}$. However, we show (see Theorem 40) that the algorithm has the property that

$$\sum_{t=1}^{T} \|\Gamma_{t+1} - P_t + \hat{A}_{t+1}\hat{X}_t\|_2^2 = O(\sqrt{T}),$$

which implies that over time, on average, the equality constraint (10.4) gets satisfied. The main results is summarized in the following theorem:

Theorem 39 *Let $\{\Gamma_t, \hat{A}_t, \Delta_t\}$ be the sequences generated by the OIADMM procedure and $R(T)$ be the regret as defined above. Assume the following conditions hold: (1) $\forall t$, the Frobenius norm of $\partial\|\Gamma_t\|_1$ is upper bounded by Φ; (2) $\hat{A}_1 = \mathbf{0}_{m\times k}, \|A^{\text{opt}}\|_F \leq D$; (3) $\Delta_1 = \mathbf{0}_{m\times n}$; (4) $\forall t$, $1/\tau_t \geq 2\Psi_{\max}(\hat{X}_t)$. Setting $\forall t$, $\beta_t = \frac{\Phi}{D}\sqrt{\tau_m T}$ where $\tau_m = \max_t \{1/\tau_t\}$, we have*

$$R(T) \leq \frac{\Phi D\sqrt{T}}{\sqrt{\tau_m}} + \sum_{t=1}^{T} \|A^{\text{opt}} E_t\|_1.$$

In the above theorem one could replace τ_m by any upper bound on it (i.e., we don't need to know τ_m exactly).

Condition on E_t's for Sublinear Regret. In a standard online learning setting, the $(P_t, \hat{X}_t)$ made available to the online learning algorithm will be the same as (P_t, X_t) made available to the batch dictionary learning algorithm in hindsight, so that $\hat{X}_t = X_t \Rightarrow E_t = \mathbf{0}$, yielding a $O(\sqrt{T})$ regret. More generally, as long as

$$\sum_{t=1}^{T} \|E_t\|_p = o(T)$$

for some suitable p-norm, we get a sublinear regret bound.[5] For example, if $\{Z_t\}$ is a sequence of matrices such that for all t, $\|Z_t\|_p = O(1)$, then setting $E_t = t^{-\epsilon} Z_t, \epsilon > 0$ yields a sublinear regret. This gives a sufficient condition for sublinear regret[6] and it is an interesting open problem to extend the analysis to other cases.

As mentioned in Section 10.4.1, OIADMM can violate the equality constraint at each t (i.e., $P_t - \hat{A}_{t+1}\hat{X}_t \neq \Gamma_{t+1}$). However, we show in Theorem 40 that the accumulated loss caused by the violation of equality constraint is sublinear in T, i.e., the equality constraint is satisfied on average in the long run.

Theorem 40 *Let $\{\Gamma_t, \hat{A}_t, \Delta_t\}$ be the sequences generated by the OIADMM procedure. Assume the following conditions hold: (1), $\forall t$, the Frobenius norm of $\partial\|\Gamma_t\|_1$ is upper bounded by Φ; (2), $\hat{A}_1 = \mathbf{0}_{m\times k}, \|A^{\text{opt}}\|_F \leq D$,*

[5] This follows from Hölder's inequality which gives $\sum_{t=1}^{T} \|A^{\text{opt}} E_t\|_1 \leq \|A^{\text{opt}}\|_q (\sum_{t=1}^{T} \|E_t\|_p)$ for $1 \leq p, q \leq \infty$ and $1/p + 1/q = 1$, and by the assuming $\|A^{\text{opt}}\|_q$ is bounded. Here, $\|\cdot\|_p$ denotes Schatten p-norm.

[6] In a different context, a similar assumption on the rate of error decay appeared in a recent chapter by Schmidt *et al.* [173] while analyzing the convergence rates of inexact proximal gradient methods.

$\Delta_1 = \mathbf{0}_{m\times n}$*; (3),* $\forall t$, $1/\tau_t \geq 2\Psi_{\max}(\hat{X}_t)$*. Setting* $\forall t$, $\beta_t = \frac{\Phi}{D}\sqrt{\tau_m T}$ *where* $\tau_m = \max_t \{1/\tau_t\}$*, we have*

$$\sum_{t=1}^{T} \|\Gamma_{t+1} - P_t + \hat{A}_{t+1}\hat{X}_t\|_2^2 \leq \frac{2D^2}{\tau_m} + \frac{4\Upsilon D\sqrt{T}}{\Phi\sqrt{\tau_m}}.$$

Again, as was the case with Theorem 39, we could replace τ_m in the above theorem by any upper bound on it.

Running Time. For the ith column in the dictionary matrix the projection onto $\mathcal{A}$ can be done in $O(s_i \log m)$ time where s_i is the number of non-zero elements in the ith column using the projection onto ℓ_1-ball algorithm of Duchi *et al.* [51]. The simplest implementation of OIADMM takes $O(mnk)$ time at each timestep because of the matrix multiplications involved. However, in practice, we can exploit the sparsities of the matrices to make the algorithm run much faster. OIADMM is also memory efficient, as at each time t, other than the current iterates, it only need $\hat{A}_{t-1}$ from previous timesteps.

10.5 Experimental Results

In this section, we present experiments to compare and contrast the performance of ℓ_1-batch and ℓ_1-online dictionary learning algorithms for the task of novel document detection. We also present results highlighting the superiority of using an ℓ_1- over an ℓ_2-penalty on the reconstruction error for this task (validating the discussion in Section 10.3).

Implementation of 10.3. In our implementation, we grow the dictionary size by η in each timestep. Growing the dictionary size is essential for the batch algorithm because as t increases the number of columns of $P_{[t]}$ also increases, and therefore, a larger dictionary is required to compactly represent all the documents in $P_{[t]}$. For solving (10.3), we use alternative minimization over the variables. The complete pseudo-code is given Algorithm (12) (see Appendix 10.D). The optimization problems arising in the sparse coding and dictionary learning steps are solved using ADMM's.

Online Algorithm for Novel Document Detection. Our online algorithm (Algorithm (9))[7] uses the same novel document detection step as Algorithm 10.3, but dictionary learning is done using OIADMM.

[7] In our experiments, the number of documents introduced in each timestep is almost of the same order, and hence there is no need to change the size of the dictionary across timesteps for the online algorithm.

[8] Before invoking Algorithm OIADMM we may have to zero-pad the matrices in the arguments appropriately.

Algorithm 9 : ONLINE

```
1: Input: $P_t = [\mathbf{p}_1, \ldots, \mathbf{p}_{n_t}] \in \mathbb{R}^{m \times n_t}$, $\hat{A}_t \in \mathbb{R}^{m \times k}$, $\Delta_t \in \mathbb{R}^{m \times n_t}$, $\lambda \geq 0$, $\zeta \geq 0$, $\beta \geq 0$,
   $\tau \geq 0$
2: Novel Document Detection Step:
3: for $j = 1$ to $n_t$ do
4:     Solve: $\mathbf{x}_j = \text{argmin}_{\mathbf{x} \geq 0} \|\mathbf{p}_j - \hat{A}_t \mathbf{x}\|_1 + \lambda \|\mathbf{x}\|_1$
5:     if $\|\mathbf{p}_j - \hat{A}_t \mathbf{x}_j\|_1 + \lambda \|\mathbf{x}_j\|_1 > \zeta$
6:         Mark $\mathbf{p}_j$ as novel
7: Online Dictionary Learning Step:
8: Set $\hat{X}_t \longleftarrow [\mathbf{x}_1, \ldots, \mathbf{x}_{n_t}]$
9: $(\hat{A}_{t+1}, \Delta_{t+1}) \longleftarrow$ OIADMM$(P_t, \hat{A}_t, \Delta_t, \hat{X}_t, \beta, \tau)$[8]
```

Notice that the sparse coding matrices of the Algorithm 10.3, $X_1, \ldots, X_t$ could be different from $\hat{X}_1, \ldots, \hat{X}_t$. If these sequence of matrices are close to each other, then we have a sublinear regret on the objective function.[9]

Evaluation of Novel Document Detection. For performance evaluation, we assume that documents in the corpus have been manually identified with a set of topics. For simplicity, we assume that each document is tagged with the single, most dominant topic that it associates with, which we call the *true topic* of that document. We call a document $\mathbf{y}$ arriving at time t *novel* if the true topic of $\mathbf{y}$ has not appeared before the time t. So at time t, given a set of documents, the task of novel document detection is to classify each document as either novel (positive) or non-novel (negative). For evaluating this classification task, we use the standard Area Under the ROC Curve (AUC) [130].

Performance Evaluation for ℓ_1-Dictionary Learning. We use a simple reconstruction error measure for comparing the dictionaries produced by our ℓ_1-batch and ℓ_1-online algorithms. We want the dictionary at time t to be a good basis to represent all the documents in $P_{[t]} \in \mathbb{R}^{m \times N_t}$. This leads us to define the *sparse reconstruction error* (SRE) of a dictionary A at time t as

$$\text{SRE}(A) \overset{\text{def}}{=} \frac{1}{N_t} \left(\min_{X \geq \mathbf{0}} \|P_{[t]} - AX\|_1 + \lambda \|X\|_1 \right).$$

A dictionary with a smaller SRE is better on average at sparsely representing the documents in $P_{[t]}$.

[9] As noted earlier, we can not do a comparison without making any assumptions.

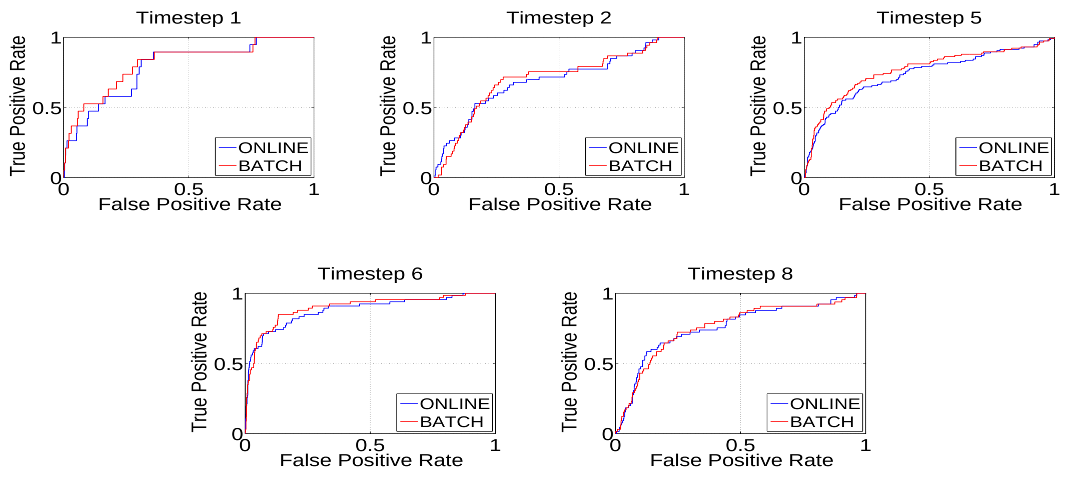


Figure 10.1: ROC curves for TDT2 for timesteps where novel documents were introduced.

Novel Document Detection using ℓ_2-dictionary learning. To justify the choice of using an ℓ_1-penalty (on the reconstruction error) for novel document detection, we performed experiments comparing ℓ_1- vs. ℓ_2-penalty for this task. In the ℓ_2-setting, for the sparse coding step we used a fast implementation of the LARS algorithm with positivity constraints [62] and the dictionary learning was done by solving a non-negative matrix factorization problem with additional sparsity constraints (also known as the non-negative sparse coding problem [87]). A pseudo-code description is given in Appendix 10.D.[10]

Experimental Setup. All reported results are based on a Matlab implementation running on a quad-core 2.33 GHz Intel processor with 32GB RAM. The parameters to our ℓ_1-online dictionary learning algorithm are: (1), initial size of dictionary; (2), regularization parameter; (3), parameters to OIADMM (β_t and τ_t); (4), ADMM parameters for sparse coding. The ℓ_1-batch and ℓ_2-batch dictionary learning algorithm take an additional parameter η which describes the increase in the batch dictionary size in each timestep. The regularization parameter λ is set to 0.1 which yields reasonable sparsities in our experiments. OIADMM parameters τ_t is set $1/(2\Psi_{\max}(\hat{X}_t))$ (chosen according to Theorem 39) and β_t is fixed to 5 (obtained through tuning). The ADMM parameters for sparse coding and batch dictionary learning are set as suggested in [97] (see Appendix app:admm). In the batch algorithms, we grow the dictionary sizes

[10] We used the SPAMS package `http://spams-devel.gforge.inria.fr/` in our implementation.

by $\eta = 10$ in each timestep. The threshold value ζ is treated as a tunable parameter.

10.5.1 Experiments on News Streams

Our first dataset is drawn from the NIST Topic Detection and Tracking (TDT2) corpus which consists of news stories in the first half of 1998. In our evaluation, we used a set of 9000 documents represented over 19528 terms and distributed into the top 30 TDT2 human-labeled topics over a period of 27 weeks. We introduce the documents in groups. At timestep 0, we introduce the first 1000 documents and these documents are used for initializing the dictionary. We use an alternative minimization procedure over the variables of (10.1) to initialize the dictionary. In these experiments the size of the initial dictionary $k = 200$. In each subsequent timestep $t \in \{1, \dots, 8\}$, we provide the batch and online algorithms the same set of 1000 documents. In Figure 10.1, we present novel document detection results for those timesteps where at least one novel document was introduced. Table 10.1 shows the corresponding AUC numbers. The results show that using an ℓ_1-penalty on the reconstruction error is better for novel document detection than using an ℓ_2-penalty.

Table 10.1: AUC Numbers for ROC Plots in Figure 10.1.

Timestep	*No. of Novel Docs.*	*No. of Nonnovel Docs.*	*AUC ℓ_1-online*	*AUC ℓ_1-batch*	*AUC ℓ_2-batch*
1	19	981	0.791	0.815	0.674
2	53	947	0.694	0.704	0.586
5	116	884	0.732	0.764	0.601
6	66	934	0.881	0.898	0.816
8	65	935	0.757	0.760	0.701
Avg.			**0.771**	**0.788**	**0.676**

Comparison of the ℓ_1-online and ℓ_1-batch Algorithms. The ℓ_1-online and ℓ_1-batch algorithms have almost identical performance in terms of detecting novel documents (see Table 10.1). However, the online algorithm is much more computationally efficient. In Figure 10.2(a), we compare the running times of these algorithms. As noted earlier, the running time of the batch algorithm goes up as t increases (as it has to optimize over the entire past). However, the running time of the online algorithm is independent of the past and only depends on the number of documents introduced in each timestep (which in this case is always 1000). Therefore, the running time of the online algorithm is almost the same across different timesteps. As expected

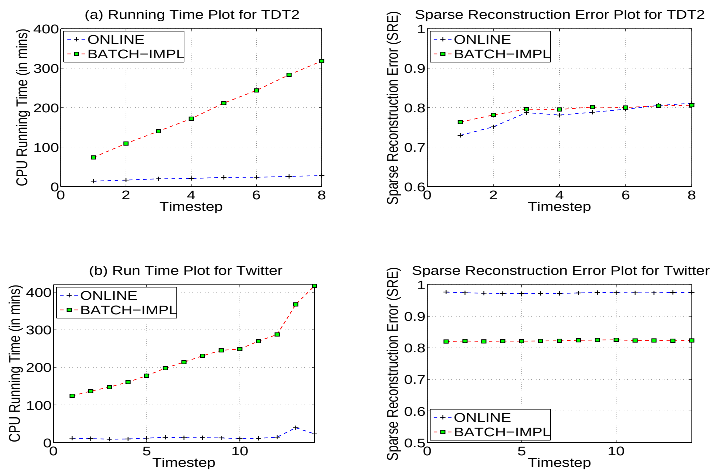


Figure 10.2: Running time and SRE plots for TDT2 and Twitter datasets.

the run-time gap between the ℓ_1-batch and ℓ_1-online algorithms widen as t increases – in the first timestep the online algorithm is 5.4 times faster, and this rapidly increases to a factor of 11.5 in just 7 timesteps.

In Figure 10.2(b), we compare the dictionaries produced by the ℓ_1-batch and ℓ_1-online algorithms under the SRE metric. In the first few timesteps, the SRE of the dictionaries produced by the online algorithm is slightly lower than that of the batch algorithm. However, this gets corrected after a few timesteps and as expected later on the batch algorithm produces better dictionaries.

10.5.2 Experiments on Twitter

Our second dataset is from an application of monitoring Twitter for Marketing and PR for smartphone and wireless providers. We used the Twitter Decahose[7] to collect a 10% sample of all tweets (posts) from Sept 15 to Oct 05, 2011. From this, we filtered the tweets relevant to "Smartphones" using a scheme presented in [35] which utilizes the Wikipedia ontology to do the filtering. Our dataset comprises of 127760 tweets over these 21 days and the vocabulary size is 6237 words. We used the tweets from Sept 15 to 21 (34292 in number) to initialize the dictionaries. Subsequently, at each timestep, we give as input to both the algorithms all

[7] htpp://gnip.com/twitter/decahose

the tweets from a given day (for a period of 14 days between Sept 22 to Oct 05). Since this dataset is unlabeled, we do a quantitative evaluation of ℓ_1-batch vs. ℓ_1-online algorithms (in terms of SRE) and do a qualitative evaluation of the ℓ_1-online algorithm for the novel document detection task. Here, the size of the initial dictionary $k = 100$.

Figure 10.2(c) shows the running times on the Twitter dataset. At first timestep the online algorithm is already 10.8 times faster, and this speedup escalates to 18.2 by the 14th timestep. Figure 10.2(d) shows the SRE of the dictionaries produced by these algorithms. In this case, the SRE of the dictionaries produced by the batch algorithm is consistently better than that of the online algorithm, but as the running time plots suggests this improvement comes at a very steep price.

Table 10.2 below shows a representative set of novel tweets identified by our online algorithm. In each timestep, instead of thresholding by ζ, we take the top 10% of tweets measured in terms of the sparse coding objective value and run a dictionary-based clustering, described in [97], on it. Further post-processing is done to discard clusters without much support and to pick a representative tweet for each cluster. Using this completely automated process, we are able to detect breaking news and trends relevant to the smartphone market, such as AT&T throttling data bandwidth, launch of IPhone 4S, and the death of Steve Jobs.

Date	*Sample Novel Tweets Detected Using our Online Algorithm*
2011-09-26	Android powered 56 percent of smartphones sold in the last three months. Sad thing is it can't lower the rating of ios!
2011-09-29	How Windows 8 is faster, lighter and more efficient: WP7 Droid Bionic Android 2.3.4 HP TouchPad white ipods_72
2011-10-03	U.S. News: AT&T begins sending throttling warnings to top data hogs: AT&T did away with its unlimited da... #iPhone
2011-10-04	Can't wait for the iphone 4s #Let ustalkiphone
2011-10-05	Everybody put an iPhone up in the air one time #ripstevejobs

Table 10.2: Sample novel documents detected by our online algorithm.

Appendix

10.A Proof of Theorem 39

First, Let us recap the OIADMM update rules.

$$\Gamma_{t+1} = \underset{\Gamma}{\operatorname{argmin}} \, \|\Gamma\|_1 + \langle \Delta_t, \widetilde{\Gamma}_t - \Gamma \rangle + \frac{\beta_t}{2} \|\widetilde{\Gamma}_t - \Gamma\|_F^2 \, , \tag{10.7}$$

$$\hat{A}_{t+1} = \operatorname*{argmin}_{A \in \mathcal{A}} \beta_t(\langle G_{t+1}, A - \hat{A}_t \rangle + \frac{1}{2\tau_t}\|A - \hat{A}_t\|_F^2), \tag{10.8}$$

$$\Delta_{t+1} = \Delta_t + \beta_t(P_t - \hat{A}_{t+1}X_t - \Gamma_{t+1}). \tag{10.9}$$

Let A^{opt} be the optimum solution to (the batch problem)

$$\min_{A \in \mathcal{A}} \sum_{t=1}^{T} \|P_t - AX_t\|_1.$$

Let $\widetilde{\Gamma}_t = P_t - \hat{A}_t\hat{X}_t$ and $\widehat{\Gamma}_t = P_t - \hat{A}_{t+1}\hat{X}_t$. For any, $A^\star \in \mathcal{A}$, let $\Gamma_t^\star = P_t - A^\star\hat{X}_t$. The lemmas below hold for any $A^\star \in \mathcal{A}$ so in particular it holds for $A^\star$ set as A^{opt}.

Proof Flow. Although the algorithm is relatively simple, the analysis is somewhat involved. Define, $\overline{\Gamma}_t^{\text{opt}} = P_t - A^{\text{opt}}X_t$. Then the regret of the OIADMM is

$$R(T) = \sum_{t=1}^{T} \|\widetilde{\Gamma}_t\|_1 - \|\overline{\Gamma}_t^{\text{opt}}\|_1.$$

We split the proof into three technical lemmas. We first upper bound $\langle \Delta_t, \widehat{\Gamma}_t - \Gamma_t^\star \rangle$ (Lemma 36), and use it to bound $\|\Gamma_{t+1}\|_1 - \|\Gamma_t^\star\|_1$ (Lemma 37). In the proof of Lemma 38, we bound $\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_{t+1}\|_1$ and this when added to the bound on $\|\Gamma_{t+1}\|_1 - \|\Gamma_t^\star\|_1$ (from Lemma 37) gives a bound on $\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^\star\|_1$. The proof of the regret bound uses a canceling telescoping sum on the bound on $\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^\star\|_1$.

We use the following simple inequality in our proofs.

Lemma 35 *For matrices $M_1, M_2, M_3, M_4 \in \mathbb{R}^{m \times n}$, we have the following*

$$2\langle M_1 - M_2, M_3 - M_4 \rangle = \|M_1 - M_4\|_F^2 + \|M_2 - M_3\|_F^2 - \|M_1 - M_3\|_F^2 - \|M_2 - M_4\|_F^2.$$

Lemma 36 *Let $\{\Gamma_t, \hat{A}_t, \Delta_t\}$ be the sequences generated by the OIADMM procedure. For any $A^\star \in \mathcal{A}$, we have*

$$\langle \Delta_t, \widehat{\Gamma}_t - \Gamma_t^\star \rangle \leq \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right)$$
$$+\frac{\beta_t}{2}\left(\|\Gamma_t^\star - \Gamma_{t+1}\|_F^2 - \|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 - \|\Gamma_t^\star - \widetilde{\Gamma}_t\|_F^2\right) - \frac{\beta_t}{2}\left(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2.$$

Proof: For any $A^\star \in \mathcal{A}$, (10.8) is equivalent to the following variational inequality [167]:

$$\beta_t\langle G_{t+1} + \frac{1}{\tau_t}(\hat{A}_{t+1} - \hat{A}_t), A^\star - \hat{A}_{t+1} \rangle \geq 0. \tag{10.10}$$

Using $\widehat{\Gamma}_t = P_t - \hat{A}_{t+1}\hat{X}_t$ and substituting for G_{t+1}, we have

$$\begin{aligned}\beta_t\langle G_{t+1}, A^\star - \hat{A}_{t+1}\rangle &= -\beta_t\langle(\Delta_t/\beta_t + \widetilde{\Gamma}_t - \Gamma_{t+1})\hat{X}_t^\top, A^\star - \hat{A}_{t+1}\rangle \\ &= \beta_t\langle\Delta_t/\beta_t + \widetilde{\Gamma}_t - \Gamma_{t+1}, \hat{A}_{t+1}\hat{X}_t - A^\star\hat{X}_t\rangle \\ &= \beta_t\langle\Delta_t/\beta_t + \widetilde{\Gamma}_t - \Gamma_{t+1}, P_t - A^\star\hat{X}_t - (P_t - \hat{A}_{t+1}\hat{X}_t)\rangle \\ &= \langle\Delta_t, \Gamma_t^\star - \widehat{\Gamma}_t\rangle + \beta_t\langle\widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_t^\star - \widehat{\Gamma}_t\rangle. \qquad (10.11)\end{aligned}$$

Substituting (10.11) into (10.10) and rearranging the terms yield

$$\langle\Delta_t, \widehat{\Gamma}_t - \Gamma_t^\star\rangle \le \beta_t\langle\widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_t^\star - \widehat{\Gamma}_t\rangle + \frac{\beta_t}{\tau_t}\langle\hat{A}_{t+1} - \hat{A}_t, A^\star - \hat{A}_{t+1}\rangle. \qquad (10.12)$$

By using Lemma 35, the first term on the right side can be rewritten as

$$\langle\widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_t^\star - \widehat{\Gamma}_t\rangle = \frac{1}{2}\big(\|\widetilde{\Gamma}_t - \widehat{\Gamma}_t\|_F^2 + \|\Gamma_t^\star - \Gamma_{t+1}\|_F^2 - \|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 - \|\Gamma_t^\star - \widetilde{\Gamma}_t\|_F^2\big). \qquad (10.13)$$

Substituting the definitions of $\widehat{\Gamma}_t$ and $\widetilde{\Gamma}_t$, we have

$$\|\widetilde{\Gamma}_t - \widehat{\Gamma}_t\|_F^2 = \|P_t - \hat{A}_t\hat{X}_t - (P_t - \hat{A}_{t+1}\hat{X}_t)\|_F^2 = \|(\hat{A}_{t+1} - \hat{A}_t)\hat{X}_t\|_F^2 \le \Psi_{\max}(\hat{X}_t)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2, \qquad (10.14)$$

Remember that $\Psi_{\max}(\hat{X}_t)$ is the maximum eigenvalue of $X^\top X$. Using Lemma 35, we get that the second term in the right hand side of (10.12) is equivalent to

$$\langle\hat{A}_{t+1} - \hat{A}_t, A^\star - \hat{A}_{t+1}\rangle = \frac{1}{2}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2 - \|\hat{A}_{t+1} - \hat{A}_t\|_F^2\right). \qquad (10.15)$$

Combining results in (10.12), (10.13), (10.14), and (10.15), we get the desired bound. ∎

Lemma 37 *Let $\{\Gamma_t, \hat{A}_t, \Delta_t\}$ be the sequences generated by the OIADMM procedure. For any $A^\star \in \mathcal{A}$, we have*

$$\begin{aligned}\|\Gamma_{t+1}\|_1 - \|\Gamma_t^\star\|_1 \le \frac{1}{2\beta_t}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right) \\ - \frac{\beta_t}{2}\left(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2 - \frac{\beta_t}{2}\|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2.\end{aligned}$$

Proof: Let $\partial\|\Gamma_{t+1}\|_1$ denote the subgradient of $\|\Gamma_{t+1}\|_1$. Now Γ_{t+1} is a minimizer of (10.7). Therefore, $\mathbf{0}_{m\times n} \in \partial\|\Gamma_{t+1}\|_1 - \Delta_t - \beta_t(\widetilde{\Gamma}_t - \Gamma_{t+1})$. Rearranging the terms gives $\Delta_t + \beta_t(\widetilde{\Gamma}_t - \Gamma_{t+1}) \in \partial\|\Gamma_{t+1}\|_1$. Since $\|\Gamma_{t+1}\|_1$ is a convex function, we have

$$\begin{aligned}\|\Gamma_{t+1}\|_1 - \|\Gamma_t^\star\|_1 &\le \langle \Delta_t + \beta_t(\widetilde{\Gamma}_t - \Gamma_{t+1}), \Gamma_{t+1} - \Gamma_t^\star\rangle \\ &\le \langle \Delta_t, \Gamma_{t+1} - \widehat{\Gamma}_t\rangle + \langle \Delta_t, \widehat{\Gamma}_t - \Gamma_t^\star\rangle + \beta_t\langle \widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_{t+1} - \Gamma_t^\star\rangle. \quad (10.16)\end{aligned}$$

Using Lemma 35, the last term can be rewritten as

$$\beta_t\langle \widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_{t+1} - \Gamma_t^\star\rangle = \frac{\beta_t}{2}\left(\|\Gamma_t^\star - \widetilde{\Gamma}_t\|_F^2 - \|\Gamma_t^\star - \Gamma_{t+1}\|_F^2 - \|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2\right) \quad (10.17)$$

Combining the inequality of Lemma 36 with (10.17) gives

$$\begin{aligned}\langle \Delta_t, \widehat{\Gamma}_t - \Gamma_t^\star\rangle + \beta_t\langle \widetilde{\Gamma}_t - \Gamma_{t+1}, \Gamma_{t+1} - \Gamma_t^\star\rangle &\le \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right) \\ - \frac{\beta_t}{2}\left(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2 &- \frac{\beta_t}{2}(\|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2 - \|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2). \quad (10.18)\end{aligned}$$

Since $\Gamma_{t+1} - \widehat{\Gamma}_t = (\Delta_t - \Delta_{t+1})/\beta_t$, we have

$$\begin{aligned}\langle \Delta_t, \Gamma_{t+1} - \widehat{\Gamma}_t\rangle - \frac{\beta_t}{2}\|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 &= \frac{1}{2\beta_t}\left(2\langle \Delta_t, \Delta_t - \Delta_{t+1}\rangle - \|\Delta_t - \Delta_{t+1}\|_F^2\right) \\ &= \frac{1}{2\beta_t}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right). \quad (10.19)\end{aligned}$$

Plugging (10.18) and (10.19) into (10.16) yields the result. ■

Lemma 38 *Let $\{\Gamma_t, \hat{A}_t, \Delta_t\}$ be the sequences generated by the OIADMM procedure. If τ_t satisfies $\frac{1}{\tau_t} \ge 2\Psi_{\max}(\hat{X}_t)$. Then*

$$\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^\star\|_1 \le \frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{1}{2\beta_t}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right),$$

where $\Lambda_t \in \partial\|\widetilde{\Gamma}_t\|_1$.

Proof: Let $\Lambda_t \in \partial\|\widetilde{\Gamma}_t\|_1$. Therefore, $\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_{t+1}\|_1 \le \langle \Lambda_t, \widetilde{\Gamma}_t - \Gamma_{t+1}\rangle$. Now,

$$\langle \Lambda_t, \widetilde{\Gamma}_t - \Gamma_{t+1}\rangle = \langle \Lambda_t/\sqrt{\beta_t}, \sqrt{\beta_t}(\widetilde{\Gamma}_t - \Gamma_{t+1})\rangle \le \frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{\beta_t}{2}\|\widetilde{\Gamma}_t - \Gamma_{t+1}\|_F^2$$

Therefore,

$$\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_{t+1}\|_1 \le \frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{\beta_t}{2}\|\widetilde{\Gamma}_t - \Gamma_{t+1}\|_F^2. \quad (10.20)$$

Adding (10.20) and the inequality of Lemma 37 together we get

$$\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^\star\|_1 \leq \frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{1}{2\beta_t}(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2) + \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right) - \frac{\beta_t}{2}\left(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2.$$

Setting $1/\tau_t \geq 2\Psi_{\max}(\hat{X}_t)$ means that $(-\beta_t/2)(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t))\|\hat{A}_{t+1} - \hat{A}_t\|_F^2 \leq 0$, Therefore,

$$\|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^\star\|_1 \leq \frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{1}{2\beta_t}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{\beta_t}{2\tau_t}\left(\|A^\star - \hat{A}_t\|_F^2 - \|A^\star - \hat{A}_{t+1}\|_F^2\right),$$

■

Now, we are ready to prove Theorem 39. *Proof:* Substituting, $\Gamma_t^{\text{opt}} = P_t - A^{\text{opt}}\hat{X}_t$ for $\Gamma_t^\star$ and A^{opt} for $A^\star$ in Lemma 38. Set $\beta_t = \frac{\Phi}{D}\sqrt{\tau_m T}$.

$$\begin{aligned}
&\sum_{t=1}^{T} \|\widetilde{\Gamma}_t\|_1 - \|\Gamma_t^{\text{opt}}\|_1 \\
&\leq \sum_{t=1}^{T}\left(\frac{1}{2\beta_t}\|\Lambda_t\|_F^2 + \frac{1}{2\beta_t}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{\beta_t}{2\tau_t}\left(\|A^{\text{opt}} - \hat{A}_t\|_F^2 - \|A^{\text{opt}} - \hat{A}_{t+1}\|_F^2\right)\right) \\
&\leq \frac{D}{2\Phi\sqrt{\tau_m T}}\sum_{t=1}^{T}\|\Lambda_t\|_F^2 + \frac{D}{2\Phi\sqrt{\tau_m T}}\sum_{t=1}^{T}(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2) \\
&+ \frac{\Phi\sqrt{T}}{2D\sqrt{\tau_m}}\sum_{t=1}^{T}(\|A^{\text{opt}} - \hat{A}_t\|_F^2 - \|A^{\text{opt}} - \hat{A}_{t+1}\|_F^2) \\
&\leq \frac{D}{2\Phi\sqrt{\tau_m T}}\cdot(T\Phi^2) + \frac{D}{2\Phi\sqrt{\tau_m T}}\cdot(\|\Delta_1\|_F^2) + \frac{\Phi\sqrt{T}}{2D\sqrt{\tau_m}}\cdot\|A^{\text{opt}} - \hat{A}_1\|_F^2 \\
&\leq \frac{D\sqrt{T}\Phi}{2\sqrt{\tau_m}} + 0 + \frac{D\sqrt{T}\Phi}{2\sqrt{\tau_m}} \\
&= \frac{D\sqrt{T}\Phi}{\sqrt{\tau_m}}.
\end{aligned}$$

Since

$$\overline{\Gamma}_t^{\text{opt}} = P_t - A^{\text{opt}}X_t = P_t - A^{\text{opt}}(\hat{X}_t + E_t) = \Gamma_t^{\text{opt}} - A^{\text{opt}}E_t,$$

we have then $\|\overline{\Gamma}_t^{\text{opt}}\|_1 + \|A^{\text{opt}}E_t\|_1 \geq \|\Gamma_t^{\text{opt}}\|_1$. The regret is bounded as follows:

$$R(T) = \sum_{t=1}^{T}\|\widetilde{\Gamma}_t\|_1 - \|\overline{\Gamma}_t^{\text{opt}}\|_1 \leq \frac{\Phi D\sqrt{T}}{\sqrt{\tau_m}} + \sum_{t=1}^{T}\|A^{\text{opt}}E_t\|_1.$$

■

10.B Proof of Theorem 40

Proof: Let $\widehat{\Gamma}_t = P_t - \hat{A}_{t+1}\hat{X}_t$. Let us look at $\|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2$.

$$\begin{aligned}\|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 = \|\Gamma_{t+1} - \widetilde{\Gamma}_t + \widetilde{\Gamma}_t - \widehat{\Gamma}_t\|_F^2 &\leq 2\left(\|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2 + \|\widetilde{\Gamma}_t - \widehat{\Gamma}_t\|_F^2\right) \\ &\leq 2\left(\|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2 + \Psi_{\max}(\hat{X}_t)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2\right). \qquad (10.21)\end{aligned}$$

For the first inequality, we used the simple fact that for any two matrices M_1 and M_2 $\|M_1 - M_2\|_F^2 \leq 2(\|M_1\|_F^2 + \|M_2\|_F^2)$. The second inequality is because of (10.14). Firstly, since $\|\Gamma_{t+1}\|_1 \geq 0$

$$\|\Gamma_{t+1}\|_1 - \|\Gamma_t^{\text{opt}}\|_1 \geq -\|\Gamma_t^{\text{opt}}\|_1 \geq -\Upsilon.$$

Using this and rearranging terms in the inequality of Lemma 37 (with A^{opt} instead of $A^\star$) gives

$$\begin{aligned}\|\Gamma_{t+1} - \widetilde{\Gamma}_t\|_F^2 \leq \frac{1}{\beta_t^2}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{1}{\tau_t}\left(\|A^{\text{opt}} - \hat{A}_t\|_F^2 - \|A^{\text{opt}} - \hat{A}_{t+1}\|_F^2\right) \\ - \left(\frac{1}{\tau_t} - \Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2 + \frac{2\Upsilon}{\beta_t},\end{aligned}$$

Plugging this into (10.21) yields

$$\begin{aligned}\|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 \leq \frac{2}{\beta_t^2}\left(\|\Delta_t\|_F^2 - \|\Delta_{t+1}\|_F^2\right) + \frac{2}{\tau_t}\left(\|A^{\text{opt}} - \hat{A}_t\|_F^2 - \|A^{\text{opt}} - \hat{A}_{t+1}\|_F^2\right) \\ - 2\left(\frac{1}{\tau_t} - 2\Psi_{\max}(\hat{X}_t)\right)\|\hat{A}_{t+1} - \hat{A}_t\|_F^2 + \frac{4\Upsilon}{\beta_t}.\end{aligned}$$

Letting $1/\tau_t \geq 2\Psi_{\max}(\hat{X}_t)$ and summing over t from 0 to T and simplifying the resulting equation we get

$$\sum_{t=1}^{T}\|\Gamma_{t+1} - \widehat{\Gamma}_t\|_F^2 \leq \frac{2D^2}{\tau_m} + \frac{4\Upsilon D\sqrt{T}}{\Phi\sqrt{\tau_m}}.$$

■

10.C ADMM Equations for updating X and A's

Consider the ℓ_1-dictionary learning problem

$$\min_{A\in\mathcal{A}, X\geq\mathbf{0}} \|P - AX\|_1 + \lambda\|X\|_1,$$

where $\mathcal{A}$ is defined in Section 10.2. We use the following algorithm from [97] to solve this problem. It is quite easy to adapt the ADMM updates to update X's and A's, when the other variable is fixed (see e.g., [97]).

ADMM for updating X, given fixed A. Here we are given matrices $P \in \mathbb{R}^{m\times n}$ and $A \in \mathbb{R}^{m\times k}$, and we want to solve the following optimization problem

$$\min_{X\geq \mathbf{0}} \|P - AX\|_1 + \lambda\|X\|_1 \equiv \min_{X\geq \mathbf{0},E} \|E\|_1 + \lambda\|X\|_1 \text{ such that } E = P - AX.$$

Algorithm 10 shows the ADMM update steps for solving this problem. The entire derivation is presented in [97] and we are reproducing them here for completeness. In our experiments, we set $\varphi = 5$, $\kappa = 1/\Psi_{\max}(A)$, and $\gamma = 1.89$. These parameters are chosen based on the ADMM convergence results presented in [97, 214].

Algorithm 10 : ADMM for Updating X

```
1: ADMM procedure for solving $\min_{X\geq \mathbf{0}} \|P - AX\|_1 + \lambda\|X\|_1$
2: Input: $A \in \mathbb{R}^{m\times k}$, $P \in \mathbb{R}^{m\times n}$, $\lambda \geq 0$, $\gamma \geq 0$, $\psi \geq 0$, $\kappa \geq 0$
3: $X_{(1)} \leftarrow \mathbf{0}_{k\times n}$, $E_{(1)} \leftarrow P$, $\rho_{(1)} \leftarrow \mathbf{0}_{m\times n}$
4: for $i = 1, 2, \dots,$ to convergence do
5:     $E_{(i+1)} \leftarrow \text{soft}(P - AX_{(i)} + \rho_{(i)}/\varphi, 1/\varphi)$
6:     $G \leftarrow A^\top(AX_{(i)} + E_{(i+1)} - P - \rho_{(i)}/\varphi)$
7:     $X_{(i+1)} \leftarrow \max\{X_{(i)} - \kappa G - (\lambda\kappa)/\varphi, 0\}$
8:     $\rho_{(i+1)} \leftarrow \rho_{(i)} + \gamma\varphi(P - AX_{(i+1)} - E_{(i+1)})$
9: Return $X$ at convergence
```

ADMM for Updating A, given fixed X. Given inputs $P \in \mathbb{R}^{m\times n}$ and $X \in \mathbb{R}^{k\times n}$, consider the following optimization problem

$$\min_{A\in\mathcal{A}} \|P - AX\|_1 \equiv \min_{A\in\mathcal{A},E} \|E\|_1 \text{ such that } E = P - AX.$$

When repeating this optimization over multiple timesteps, we use warm starts for faster convergence, i.e., instead of initializing $A_{(1)}$ to $\mathbf{0}_{m\times k}$, we initialize $A_{(1)}$ to the dictionary obtained at the end of the previous timestep.

Algorithm 11 : ADMM for Updating A

```
1: **ADMM procedure for solving** $\min_{A\in\mathcal{A}} \|P - AX\|_1$
2: **Input:** $X \in \mathbb{R}^{k\times n}$, $P \in \mathbb{R}^{m\times n}$, $\gamma \geq 0$, $\psi \geq 0$, $\kappa \geq 0$
3: $A_{(1)} \leftarrow \mathbf{0}_{m\times k}$, $E_{(1)} \leftarrow P$, $\rho_{(1)} \leftarrow \mathbf{0}_{m\times n}$
4: **for** $i = 1, 2, \ldots,$ **to** *convergence* **do**
5:     $E_{(i+1)} \leftarrow \text{soft}(P - A_{(i)}X + \rho_{(i)}/\varphi, 1/\varphi)$
6:     $G \leftarrow (A_{(i)}X + E_{(i+1)} - P - \rho_{(i)}/\varphi)X^\top$
7:     $A_{(i+1)} \leftarrow \Pi_{\mathcal{A}}(\max\{A_{(i)} - \kappa G, 0\})$
8:     $\rho_{(i+1)} \leftarrow \rho_{(i)} + \gamma\varphi(P - A_{(i+1)}X - E_{(i+1)})$
9: Return $A$ at convergence
```

10.D Pseudo-Codes from Section 10.5

Let us start by extending the definition of $\mathcal{A}$, define

$$\mathcal{A}_{k_t} = \{A \in \mathbb{R}^{m\times k_t} \;:\; A \geq \mathbf{0}_{m\times k_t} \;\; \forall j = 1, \ldots, k_t \;, \|A_j\|_1 \leq 1\},\text{ where } A_j \text{ is the } j\text{th column in } A.$$

We use $\Pi_{\mathcal{A}_{k_t}}$ to denote the projection onto the nearest point in the convex set $\mathcal{A}_{k_t}$.

Define $\mathbb{A}_{k_t}$ as

$$\mathbb{A}_{k_t} = \{A \in \mathbb{R}^{m\times k_t} \;:\; A \geq \mathbf{0}_{m\times k_t} \;\; \forall j = 1, \ldots, k_t \;, \|A_j\|_2 \leq 1\},\text{ where } A_j \text{ is the } j\text{th column in } A.$$

We use $\Pi_{\mathbb{A}_{k_t}}$ to denote the projection onto the nearest point in the convex set $\mathbb{A}_{k_t}$.

Algorithm 12 : BATCH-IMPL

```
1: Input: $P_{[t-1]} \in \mathbb{R}^{m \times N_{t-1}}$, $X_{[t-1]} \in \mathbb{R}^{k_t \times N_{t-1}}$, $P_t = [\mathbf{p}_1, \dots, \mathbf{p}_{n_t}] \in \mathbb{R}^{m \times n_t}$, $A_t \in \mathbb{R}^{m \times k_t}$, $\lambda, \zeta, \eta \geq 0$
2: Novel Document Detection Step:
3: for $j = 1$ to $n_t$ do
4:     Solve: $\mathbf{x}_j = \text{argmin}_{\mathbf{x} \geq 0} \|\mathbf{p}_j - A_t\mathbf{x}\|_1 + \lambda\|\mathbf{x}\|_1$     (solved using Algorithm 10)
5:     if $\|\mathbf{p}_j - A_t\mathbf{x}_j\|_1 + \lambda\|\mathbf{x}_j\|_1 > \zeta$
6:         Mark $\mathbf{p}_j$ as novel
7: Batch Dictionary Learning Step:
8: Set $k_{t+1} \leftarrow k_t + \eta$
9: Set $Z_{[t]} \leftarrow [X_{[t-1]} \,|\, \mathbf{x}_1, \dots, \mathbf{x}_{n_t}]$
10: Set $X_{[t]} \leftarrow \begin{bmatrix} Z_{[t]} \\ \mathbf{0}_{\eta \times N_t} \end{bmatrix}$
11: Set $P_{[t]} \leftarrow [P_{[t-1]} \,|\, \mathbf{p}_1, \dots, \mathbf{p}_{n_t}]$
12: for $i = 1$ to convergence do
13:     Solve: $A_{t+1} = \text{argmin}_{A \in \mathcal{A}_{k_{t+1}}} \|P_{[t]} - AX_{[t]}\|_1$     (solved using Algorithm 11 with warm starts)
14:     Solve: $X_{[t]} = \text{argmin}_{X \geq \mathbf{0}} \|P_{[t]} - A_{t+1}X\|_1 + \lambda\|X\|_1$     (solved using Algorithm 10)
```

Algorithm 13 : L2-BATCH

```
1: Input: $P_{[t-1]} \in \mathbb{R}^{m \times N_{t-1}}$, $P_t = [\mathbf{p}_1, \dots, \mathbf{p}_{n_t}] \in \mathbb{R}^{m \times n_t}$, $A_t \in \mathbb{R}^{m \times k_t}$, $\lambda \geq 0$, $\zeta \geq 0$, $\eta \geq 0$
2: Novel Document Detection Step:
3: for $j = 1$ to $n_t$ do
4:     Solve: $\mathbf{x}_j = \text{argmin}_{\mathbf{x} \geq 0} \|\mathbf{p}_j - A_t\mathbf{x}\|_2 + \lambda\|\mathbf{x}\|_1$ (solved using the LARS method [62])
5:     if $\|\mathbf{p}_j - A_t\mathbf{x}_j\|_2 + \lambda\|\mathbf{x}_j\|_1 > \zeta$
6:         Mark $\mathbf{p}_j$ as novel
7: $\ell_2$-batch Dictionary Learning Step:
8: Set $k_{t+1} \leftarrow k_t + \eta$
9: Set $P_{[t]} \leftarrow [P_{[t-1]} \,|\, \mathbf{p}_1, \dots, \mathbf{p}_{n_t}]$
10: $[A_{t+1}, X_{[t]}] = \text{argmin}_{A \in \mathbb{A}_{k_{t+1}}, X \geq \mathbf{0}} \|P_{[t]} - AX\|_2 + \lambda\|X\|_1$     (non-negative sparse coding problem)
```

Chapter 11

Conclusions

This thesis developed several novel optimization methods to address the issues encountered in large scale machine learning system, particularly for synchronization and consistency. When using the stochastic gradient descent (SGD) to solve the empirical risk minimization problems in a parameter server, it runs the risk of overwriting, which have been addressed in Part I. Part II developed several algorithms to solve equality-constrained optimization problems. Finally, we validate the effectiveness and scalability of the proposed methods in a variety of applications.

In Chaper 2, we proposed online randomized block coordinate descent (ORBCD) which combines online/stochastic gradient descent and randomized block coordinate descent. ORBCD is well suitable for large scale high dimensional problems with non-overlapping composite regularizers. We established the rate of convergence for ORBCD, which has the same order as OGD/SGD. For stochastic optimization with strongly convex functions, ORBCD can converge at a geometric rate in expectation by reducing the variance of stochastic gradient. Essentially, ORBCD updates part of model parameters using partial samples, allowing the overwriting in SGD.

In Chaper 3, we first reviewed the alternating direction method of multipliers (ADMM). We developed new proof techniques to analyze the convergence rate for ADM, which establishes a $O(1/T)$ convergence rate for the objective, the optimality conditions (constraints) and the variational inequality form of ADMM. The proof techniques facilitate the improvement and modifications of ADMM which are needed in some scenarios.

In Chapter 4, we generalized the alternating direction method of multipliers (ADMM) to Bregman ADMM, similar to how mirror descent generalizes gradient descent. BADMM defines a unified framework for ADMM, generalized ADMM, inexact ADMM and Bethe ADMM. The global convergence and the $O(1/T)$ iteration complexity of BADMM are also established. In some cases, BADMM is faster than ADMM by a factor of $O(n/\log(n))$. BADMM can also be faster than highly optimized commercial software in solving linear program of mass transportation problem.

In Chapter 5, we proposed a randomized block coordinate variant of ADMM named Parallel Direction Method of Multipliers (PDMM) to solve the class of problem of minimizing block-separable convex functions subject to linear constraints. PDMM considers the sparsity and the number of blocks to be updated when setting the step size. We show two other Jacobian ADMM methods are two special cases of PDMM. We also use PDMM to solve overlapping block problems. The global convergence and the iteration complexity are established with constant step size. Experiments on robust principal component analysis and overlapping group lasso show that PDMM is faster than existing methods.

In Chapter 6, we proposed an efficient online learning algorithm named online ADM (OADM). We established regret bounds for the objective and constraint violation for general and strongly convex functions in OADM. We also discuss inexact update to yield efficient $\mathbf{x}$ update, including mirror descent and composite objective mirror descent. Finally, we illustrate the efficacy of OADM in solving lasso and total variation problems.

In Chapter 7, we proposed a provably convergent MAP inference algorithm for large scale MRFs. The algorithm is based on the 'tree decomposition'idea from the MAP inference literature and the alternating direction method from the optimization literature. Our algorithm solves the tree structured subproblems efficiently via the sum-product algorithm and is inherently parallel. The empirical results show that the new algorithm, in its sequential version, compares favorably to other existing approximate MAP inference algorithm in terms of running time and accuracy. The experimental results on large datasets demonstrate that the parallel version scales almost linearly with the number of cores in the multi-core setting. We also implemented the algorithm using MPI and the experimental results show that our implementation scales almost linearly with the number of MPI processes for grid-structured graphs.

In Chapter 8, we presented a large scale distributed framework for the estimation of sparse precision matrix using CLIME. Our framework can scale to millions of dimensions and run on hundreds of machines. The framework is based on inexact ADMM, which decomposes the constrained optimization problem into elementary matrix multiplications and elementwise operations. Convergence rates for both the objective and optimality conditions are established. The proposed framework solves the CLIME in column-blocks and uses block cyclic distribution to achieve load balancing. We evaluate our algorithm on both shared-memory and distributed-memory architectures. Experimental results show that our algorithm is substantially more scalable than state-of-the-art methods and scales almost linearly with the number of cores.

In Chapter 9, we proposed double plugin Gaussian (DoPinG) copula estimators to deal with non-Gaussian data with missing values. DoPinG estimates the sparse precision matrix corresponding to *non-paranormal* distributions by directly estimating nonparametric correlations, including Kendall's tau and Spearman's rho. DoPinG uses two plugin procedures, leveraging existing sparse precision estimators. DoPinG consists of three steps: (1) estimate nonparametric correlations by disregarding missing values; (2) estimate the non-paranormal correlation matrix directly based on nonparametric correlations like Kendall's tau and Spearman's rho; (3) plug the estimated correlation matrix into existing sparse precision estimators to yield the sparse precision matrix. We prove that DoPinG copula estimators consistently estimate the non-paranormal correlation matrix at a rate of $O(\frac{1}{(1-\delta)}\sqrt{\frac{\log p}{n}})$, where δ is the probability of missing values. Through experiments we illustrate that by increasing number of missing values (increasing δ), the performance of the method get worse and the standard deviation is increasing in consistent with the theory. The performance of Kendall's tau and Spearman's rho is almost the same for the same percentage of missing values. Experimental results on non-Gaussian data show that DoPinG is significantly better than estimators like mGlasso, which are primarily designed for Gaussian data.

In Chapter 10, we proposed a new online ℓ_1-dictionary learning algorithm, based on which we developed a scalable approach to detecting novel documents in streams of text. We established a sublinear regret bound, and empirically demonstrate orders of magnitude speedup over the batch algorithm, without much loss in performance. A further speedup can be achieved by distributing the algorithm using known techniques. In batch setting, with the ℓ_1/ℓ_1- formulation, the dual augmented Lagrangian marginally outperforms the primal augmented Lagrangian in practice.

References

[1] M.V. Afonso, J.M. Bioucas-Dias, and M.A.T. Figueiredo. Fast image recovery using variable splitting and constrained optimization. *IEEE Transactions on Image Processing*, 19(9):2345 – 2356, 2010.

[2] M. Aharon, M. Elad, and A. Bruckstein. The k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[3] K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19:357–367, 1967.

[4] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. *Convex Optimization with Sparsity-Inducing Norms*. S. Sra, S. Nowozin, S. J. Wright., editors, Optimization for Machine Learning, MIT Press, 2011.

[5] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research (JMLR)*, 6:1705–1749, 2005.

[6] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research (JMLR)*, 9:485–516, 2008.

[7] O. Banerjee, L. E. Ghaoui, and A. dAspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 9:2261–2286, 2008.

[8] S. Barman, X. Liu, S. Draper, and B. Recht. Decomposition methods for large scale LP decoding. In *Arxiv*, 2012.

[9] P. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research (JMLR)*, 3:463–482, 2002.

[10] A. Beck and M. Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31:167–175, 2003.

[11] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Science*, 2:183202, 2009.

[12] A. Ben-Tal, T. Margalit, and A. Nemirovski. The ordered subsets mirror descent optimization method with applications to tomography. *SIAM Journal on Optimization*, 12:79–108, 2001.

[13] D. P. Bertsekas. *Consined Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[14] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[15] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice Hall, 1989.

[16] L. Blackford, J. Choi, A. Cleary, J. Demmel, I. S. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R.C. Whaley. *ScaLAPACK Users' Guide*. SIAM, 1997.

[17] D. Boley. Local linear convergence of the alternating direction method of multipliers on quadratic or linear programs. *SIAM Journal on Optimization*, 23(4):21832207, 2013.

[18] S. Boucheron, O. Bousquet, and G. Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

[19] S. Boyd, E. Chu N. Parikh, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundation and Trends Machine Learning*, 3(1):1–122, 2011.

[20] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[21] J. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for l1-regularized loss minimization. In *International Conference on Machine Learning (ICML)*, 2011.

[22] T. Cai, W. Liu, and H. Zhou. Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation. *Preprint*, 2012.

[23] T. Cai, C.H. Zhang, and H. Zhou. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *American Statistical Association*, 106:594–607, 2011.

[24] E. J. Candes, X. Li, Y. Ma, and J. Wright. Robust principal component analysis ?. *Journal of the ACM*, 58:1–37, 2011.

[25] Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1998.

[26] N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50:2050–2057, 2004.

[27] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[28] R. H. Chan, J. F. Yang, and X. M. Yuan. Alternating direction method for image inpainting in wavelet domain. *SIAM Journal on Imaging Science*, 4:807–826, 2011.

[29] V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. *Annals of Statistics*, 40:1935–1967, 2012.

[30] K.-W. Chang, C.-J. Hsieh, and C.-J. Lin. Coordinate descent method for large-scale l2-loss linear support vector machines. *Journal of Machine Learning Research (JMLR)*, 9:13691398, 2008.

[31] C. Chekuri, S. Khanna, J. Naor, and L. Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM Journal on Discrete Mathematics*, 18(3):608–625, March 2005.

[32] C. Chen, B. He, Y. Ye, and X. Yuan. The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. *Preprint*, 2013.

[33] G. Chen and M. Teboulle. Convergence analysis of a proximal-like minimization algorithm using bremgan functions. *SIAM Journal on Optimization*, 3:538–543, 1993.

[34] X. Chen, Q. Lin, S. Kim, J. G. Carbonell, and E. P. Xing. Smoothing proximal gradient method for general structured sparse regression. *The Annals of Applied Statistics*, 6:719752, 2012.

[35] V. Chenthamarakshan, P. Melville, V. Sindhwani, and R. D. Lawrence. Concept Labeling: Building Text Classifiers with Minimal Supervision. In *International Joint Conference on Artifiicial Intelligence (IJCAI)*, pages 1225–1230, 2011.

[36] J. Choi. A new parallel matrix multiplication algorithm on distributed-memory concurrent computers. In *High Performance Computing on the Information Superhighway*, 1997.

[37] P. Comtes and J. Pesquet. Proximal splitting methods in signal processsing. *Fixed-Point Algorithms for Inverse Problems in Science and Engineering Springer (Ed.)*, pages 185–212, 2011.

[38] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[39] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Methods*. Cambridge University Press, 2000.

[40] W. Dai, J. Wei, X. Zheng, J. K. Kim, S. Lee, J. Yin, Q. Ho, and E. P. Xing. Petuum: A framework for iterative-convergent distributed ML. *arXiv*, 2013.

[41] Kinderlehrer David and Stampacchia Guido. *An Introduction to Variational Inequalities and Their Applications*. Society for Industrial and Applied Mathematics, 2000.

[42] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng. Large scale distributed deep networks. In *Neural Information Processing Systems (NIPS)*, 2012.

[43] J. Dean and S. Ghemawat. Map-Reduce: simplified data processing on large clusters. In *Communications of the ACM (CACM)*, 2008.

[44] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research (JMLR)*, 13:165–202, 2012.

[45] A. P. Dempster. Covariance selection. *Biometrics*, 28(1):157–175, 1972.

[46] L. Deng and D. Yu. Deep learning: Methods and applications. *Foundations and Trends in Signal Processing*, 7(3-4):197–387, 2014.

[47] W. Deng, M. Lai, Z. Peng, and W. Yin. Parallel multi-block ADMM with $o(1/k)$ convergence. *ArXiv*, 2014.

[48] W. Deng and W. Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *ArXiv*, 2012.

[49] J. Douglas and H. H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American Mathematical Society*, 82:421–439, 1956.

[50] J. Duchi, A. Agarwal, and M. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transaction on Automatic Control*, 57:592–606, 2012.

[51] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l_1-ball for learning in high dimensions. In *International Conference on Machine Learning (ICML)*, pages 272–279, 2008.

[52] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In *Conference on Learning Theory (COLT)*, 2010.

[53] J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research (JMLR)*, 10:2873–2898, 2009.

[54] J. Eckstein and D. P. Bertsekas. An alternating direction method for linear programming. Technical report, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, 1990.

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

[56] F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*, volume I. Springer, 2003.

[57] H. Fang, K. Fang, and S. Kotz. The meta-elliptical distributions with given marginals. *Journal of Multivariate Analysis*, 82:1–16, 2002.

[58] M. A. T. Figueiredo and J. M. Bioucas-Dias. Restoration of poissonian images using alternating direction optimization. *IEEE Transactions on Image Processing*, 19:3133–3145, 2010.

[59] J. Friedman, T. Hastie, and R. Tibshirani. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Biostatistics*, 9:432–441, 2008.

[60] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[61] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and sparse group lasson. *arXiv*, 2010.

[62] Jerome Friedman, Trevor Hastie, Holger Hfling, and Robert Tibshirani. Pathwise Coordinate Optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

[63] Q. Fu, A. Banerjee, S. Liess, and P. K. Snyder. Drought detection of the last century: An MRF-based approach. In *Proceedings of the SIAM International Conference on Data Mining*, 2012.

[64] Q. Fu, H. Wang, and A. Banerjee. Bethe-ADMM for tree decomposition based parallel MAP inference. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.

[65] Q. Fu, H. Wang, A. Banerjee, S. Liess, and P. K. Snyder. MAP inference on million node graphical models: KL-divergence based alternating directions method. Technical report, Computer Science and Engineering Department, University of Minesota, 2012.

[66] D. Gabay. Applications of the method of multipliers to variational inequalities. In *Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems*. M. Fortin and R. Glowinski, eds., North-Holland: Amsterdam, 1983.

[67] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications*, 2:17–40, 1976.

[68] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, 2007.

[69] T. Goldstein, X. Bresson, and S. Osher. Geometric applications of the split Bregman method: segmentation and surface reconstruction. *Journal of Scientific Computing*, 45(1):272–293, 2010.

[70] T. Goldstein, B. Donoghue, and S. Setzer. Fast alternating direction optimization methods. *CAM report 12-35, UCLA*, 2012.

[71] G. H. Golub and C. V. Loan. *Matrix Computations*. 3rd ed. Johns Hopkins University Press, 1996.

[72] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, and C. D. Bloomfield. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, pages 531–537, 1999.

[73] K. Goto and R. Van De Geijn. High performance implementation of the level-3 BLAS. *ACM Transactions on Mathematical Software*, 35:1–14, 2008.

[74] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, 2009.

[75] E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

[76] E. Hazan, A. Kalai, S. Kale, and A. Agarwal. Logarithmic regret algorithms for online convex optimization. In *Conference on Learning Theory (COLT)*, 2006.

[77] E. Hazan and S. Kale. Projection-free online learning. In *International Conference on Machine Learning (ICML)*, 2012.

[78] B. He, M. Tao, and X. Yuan. Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM Journal of Optimization*, pages 313–340, 2012.

[79] B. He and X. Yuan. On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers. *Preprint*, 2012.

[80] B. He and X. Yuan. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50:700–709, 2012.

[81] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[82] F. L. Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of Mathematical Physics*, 20:224–230, 1941.

[83] W. Hoeffding. A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics*, 19:293–325, 1948.

[84] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

[85] M. Hong, T. Chang, X. Wang, M. Razaviyayn, S. Ma, and Z. Luo. A block successive upper bound minimization method of multipliers for linearly constrained convex optimization. *Preprint*, 2013.

[86] M. Hong and Z. Luo. On the linear convergence of the alternating direction method of multipliers. *ArXiv*, 2012.

[87] P. O. Hoyer. Non-negative sparse coding. In *IEEE Workshop on Neural Networks for Signal Processing*, pages 557–565, 2002.

[88] C. Hsieh, I. Dhillon, P. Ravikumar, and A. Banerjee. A divide-and-conquer method for sparse inverse covariance estimation. In *Neural Information Processing Systems (NIPS)*, 2012.

[89] C. Hsieh, M. Sustik, I. Dhillon, and P. Ravikumar. Sparse inverse covariance matrix estimation using quadratic approximation. In *Neural Information Processing Systems (NIPS)*, 2011.

[90] C.-J. Hsieh, K.-W. Chang, S. Keerthi C.-J. Lin, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *International Conference on Machine Learning (ICML)*, 2008.

[91] Imagenet. Large scale visual recognition challenge (ILSVRC). `http://www.image-net.org/challenges/LSVRC`.

[92] C. Jin, Q. Fu, H. Wang, A. Agrawal, W. Hendrix, W. Liao, M. Patwary, A. Banerjee, and A. Choudhary. Solving combinatorial optimization problems using relaxed linear programming: A high performance computing perspective. In *International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, 2013.

[93] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Neural Information Processing Systems (NIPS)*, 2013.

[94] V. Jojic, S. Gould, and D. Koller. Fast and smooth: Accelerated dual decomposition for MAP inference. In *Proceedings of the twenty-Seventh International Conference on Machine Learning*, 2010.

[95] A. Juditsky, A. Nemirovski, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

[96] Kaggle. The home of data science. `https://www.kaggle.com`.

[97] S. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani. Emerging topic detection using dictionary learning. In *ACM International Conference on Information and Knowledge Management (CIKM)*, pages 745–754, 2011.

[98] S. Kasiviswanathan, H. Wang, A. Banerjee, and P. Melville. Online l1-dictionary learning with application to novel document detection. In *Neural Information Processing Systems (NIPS)*, 2012.

[99] K. C. Kiwiel. Proximal minimization methods with generalized Bregman functions. *SIAM Journal on Control and Optimization*, 35:1142–1168, 1995.

[100] M. Kolar and E. Xing. Estimating sparse precision matrices from data with missing values. In *International Conference on Machine Learning (ICML)*, 2012.

[101] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[102] V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

[103] V. Koltchinskii and D. Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 2002.

[104] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531 –552, march 2011.

[105] J. Konecny and P. Richtarik. Semi-stochastic gradient descent methods. *arXiv*, 2013.

[106] W. Kruskal. Ordinal measures of association. *Journal of the American Statistical Association*, 53(284):814–861, 1958.

[107] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[108] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[109] M. Lam, E. Rothberg, and M. Wolf. The cache performance and optimization of blocked algorithms. In *Architectural Support for Programming Languages and Operating Systems*, 1991.

[110] M. Ledoux and M. Talagrand. *Probability in Banach Spaces: isoperimetry and processes*. Springer, 1991.

[111] L. Li and K.-C. Toh. An inexact interior point method for L1-reguarlized sparse covariance selection. *Mathematical Programming Computation*, 2:291–315, 2010.

[112] M. Li, D. Andersen, J. Park, A. Smola, A. Ahmed, V. Josifovski, J. Long, E. Shekita, and B. Su. Scaling distributed machine learning with the parameter server. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2014.

[113] M. Li, L. Zhou, Z. Yang, A. Li, F. Xia, D. Andersen, and A. Smola. Parameter server for distributed machine learning. In *Neural Information Processing Systems (NIPS)*, 2013.

[114] X. Li, T. Zhao, X. Yuan, and H. Liu. An R package flare for high dimensional linear regression and precision matrix estimation. *http://cran.r-project.org/web/packages/flare*, 2013.

[115] Y. Li and S. Osher. Coordinate descent optimization for ℓ_1 minimization with application to compressed sensing; a greedy algorithm. *Inverse Problems and Imaging*, 3:487503, 2009.

[116] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. *UIUC Technical Report UILU-ENG-09-2215*, 2009.

[117] R. Little and D. Rubin. *Statistical analysis with missing data*. Wiley, New York, 1987.

[118] H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman. High dimensional semiparametric gaussian copula graphical models. *The Annals of Statistics*, 40(40):2293–2326, 2012.

[119] H. Liu, J. Lafferty, and L. Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *Journal of Machine Learning Research*, 10:2295–2328, 2009.

[120] H. Liu and L. Wang. Tiger: A tuning-insensitive approach for optimally estimating Gaussian graphical models. *Preprint*, 2012.

[121] P. Loh and M. Wainwright. High-dimensional regression with noisy and missing data: Provable guarantees with non-convexity. In *Neural Information Processing Systems (NIPS)*, 2012.

[122] K. Lounici. High-dimensional covariance matrix estimation with missing observations. *ArXiv*, 2012.

[123] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. Hellerstein. Distributed graphlab: A framework for machine learning in the cloud. In *International Conference on Very Large Data Bases (VLDB)*, 2012.

[124] Z. Lu and L. Xiao. On the complexity analysis of randomized block-coordinate descent methods. *ArXiv*, 2013.

[125] Z.-Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72:735, 2002.

[126] L. Zhang M. Mahdavi and R. Jin. Mixed optimization for smooth functions. In *Neural Information Processing Systems (NIPS)*, 2013.

[127] S. Ma, L. Xue, and H. Zou. Alternating direction methods for latent variable Gaussian graphical model selection. *Neural Computation*, 25:2172–2198, 2013.

[128] M. Mahdavi, R. Jin, and T. Yang. Trading regret for efficiency: Online convex optimization with long term constraints. *Journal of Machine Learning Research (JMLR)*, 13(1):2503–2528, 2012.

[129] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.

[130] C. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[131] S. Mannor and J. N. Tsitsiklis. Online learning with constraints. In *Conference on Learning Theory (COLT)*, 2006.

[132] A. F. Martins. *The Geometry of Constrained Structured Prediction: Applications to Inference and Learning of Natural Language Syntax*. PhD thesis, Carnegie Mellon University, 2012.

[133] A. F. Martins, P. M. Aguiar, M. A. Figueiredo, N. A. Smith, and E. P. Xing. An augmented Lagrangian approach to constrained MAP inference. In *International Conference on Machine Learning (ICML)*, 2011.

[134] R. Mazumder and T. Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *Journal of Machine Learning Research*, 13:723–736, 2012.

[135] C. McDiarmid. On the method of bounded differences. pages 148–188, 1989.

[136] N. Meinshausen and P. Buhlmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

[137] P. Melville, J. Leskovec, and F. Provost. Proceedings of the first workshop on social media analytics. ACM, 2010.

[138] S. Mendelson. Rademacher averages and phase transitions in glivenko-cantelli classes. *IEEE Transactions on Information Theory*, 48(1):251–263, 2002.

[139] O. Meshi and A. Globerson. An alternating direction method for dual MAP LP relaxation. In *European Conference on Machine Learning (ECML)*, 2011.

[140] T. D. Mitchell, T. R. Carter, P. D. Jones, M. Hulme, and M. New. *A comprehensive set of high-resolution grids of monthly climate for Europe and the globe: the observed record (1901-2000) and 16 scenarios (2001-2100).* Tyndall Centre for Climate Change Research, 2004.

[141] A. Nedic and A. Ozdaglar. Cooperative distributed multi-agent optimization. In D. P. Palomar and Y. C. Eldar, editors, *Convex Optimization in Signal Processing and Communications*. Cambridge University Press, 2009.

[142] A. Nedic and A. Ozdaglar. Cooperative distributed multi-agent optimization. In *Convex Optimization in Signal Processing and Communications*. D. P. Palomar and Y. C. Eldar, eds. Cambridge University Press, 2010.

[143] A. Nemirovski. Prox-method with rate of convergence $O(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15:229–251, 2004.

[144] A. Nemirovski and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983.

[145] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, 2004.

[146] Y. Nesterov. Gradient methods for minimizing composite objective function. *Technical Report 76, Center for Operation Research and Economics (CORE), Catholic University of Louvain (UCL)*, 2007.

[147] Y. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120:221–259, 2009.

[148] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization methods. *SIAM Journal on Optimization*, 22(2):341362, 2012.

[149] M. K. Ng, P.A. Weiss, and X.M. Yuan. Solving constrained total-variation problems via alternating direction methods. *SIAM Journal on Scientific Computing*, 32(5):2710–2736, 2010.

[150] F. Niu, B. Retcht, C. Re, and S. J. Wright. Hogwild! a lock-free approach to parallelizing stochastic gradient descent. In *Neural Information Processing Systems (NIPS)*, 2011.

[151] B. Olshausen and D. Field. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

[152] H. Ouyang, N. He, L. Tran, and A. Gray. Stochastic alternating direction method of multipliers. In *International Conference on Machine Learning (ICML)*, 2014.

[153] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1:123–231, 2014.

[154] Neal Parikh and Stephen Boyd. Graph projection block splitting for distributed optimization. *Mathematical Programming Computation*, 6(1):77–102, 2014.

[155] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to twitter. In *HLT '10*, pages 181–189. Association for Computational Linguistics (ACL), 2010.

[156] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[157] R. Raina, A. Madhavan, and A. Y. Ng. Large-scale deep unsupervised learning using graphics processors. In *International Conference on Machine Learning (ICML)*, 2009.

[158] P. Ravikumar, A. Agarwal, and M. J. Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *Journal of Machine Learning Research*, 11:1043–1080, 2010.

[159] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

[160] P. Richtarik and M. Takac. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 2012.

[161] P. Richtarik and M. Takac. Parallel coordinate descent methods for big data optimization. *ArXiv*, 2013.

[162] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[163] R. Rockafellar. The multiplier method of hestenes and powell applied to convex programming. *Journal of Optimization Theory and Applications*, 12:555–562, 1973.

[164] R. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1:97–116, 1976.

[165] R. T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics. Princeton University Press, 1970.

[166] R. T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1:97–116, 1976.

[167] R. T. Rockafellar and R. J-B Wets. *Variational Analysis*. Springer-Verlag, 2004.

[168] N. Le Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Neural Information Processing Systems (NIPS)*, 2012.

[169] L. Rudin, S. J. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[170] A. Saha and V. Sindhwani. Learning evolving and emerging topics in social media: A dynamic nmf approach with temporal regularization. In *ACM International Conference on Web Search and Data Mning (WSDM)*, pages 693–702, 2012.

[171] A. Saha and A. Tewari. On the non-asymptotic convergence of cyclic coordinate descent methods. *SIAM Journal on Optimization*, 23:576601, 2013.

[172] K. Scheinberg, S. Ma, and D. Goldfarb. Sparse inverse covariance selection via alternating linearization methods. In *Neural Information Processing Systems (NIPS)*, 2010.

[173] M. Schmidt, N. L. Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Neural Information Processing Systems (NIPS)*, pages 1458–1466, 2011.

[174] M. Schmidt, N. Le Roux, and F. Bach. Minimizing finite sums with the stochastic average gradient. *Technical Report HAL 00860051, INRIA, Paris, France*, 2013.

[175] S. Shalev-Shwartz, Y. Singer, and Nathan Srebro. Pegasos: primal estimated sub-gradient solver for SVM. In *International Conference on Machine Learning (ICML)*, 2007.

[176] S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1 regularized loss minimization. In *International Conference on Machine Learning (ICML)*, 2009.

[177] Shai Shalev-Shwartz. Online Learning and Online Convex Optimization. *Foundations and Trends in Machine Learning*, 4(2), 2012.

[178] N. Z. Shor. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, 1985.

[179] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In Suvrit Sra, Sebastian Nowozin, and Stephen J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.

[180] D. Sontag and T. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*.

[181] N. Stadler and P. Buhlmann. Missing values: sparse inverse covariance estimation and an extension to sparse regression. *Statistics and Computing*, pages 1–17, 2009.

[182] T. Suzuki. Dual averaging and proximal gradient descent for online alternating direction multiplier method. In *International Conference on Machine Learning (ICML)*, 2013.

[183] T. Suzuki. Stochastic dual coordinate ascent with alternating direction method of multipliers. In *International Conference on Machine Learning (ICML)*, 2014.

[184] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. 2014.

[185] M. Takac, A. Bijral, P. Richtarik, and N. Srebro. Mini-batch primal and dual methods for SVMs. In *International Conference on Machine Learning (ICML)*, 2013.

[186] R. Tappenden, P. Richtarik, and B. Buke. Separable approximations and decomposition methods for the augmented Lagrangian. *Preprint*, 2013.

[187] D. Tarlow, D. Batra, P. Kohli, and V. Kolmogorov. Dynamic tree block coordinate ascent. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning*, 2011.

[188] M. Telgarsky and S. Dasgupta. Agglomerative Bregman clustering. In *International Conference on Machine Learning (ICML)*, 2012.

[189] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

[190] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:475494, 2001.

[191] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *Preprint*, 2008.

[192] P. Tseng. Aprroximation Accuracy, Gradient Methods, and Error Bound for Structured Convex Optimization. *Mathematical Programming, Series B*, 125:263–295, 2010.

[193] H. Tsukahara. Efficient estimation in the bivariate normal copula model: Normal margins are least-favorable. *Bernoulli*, 3:55–77, 1997.

[194] H. Tsukahara. Semiparametric estimation in copula models. *Canadian Journal of Statistics*, 33:357–375, 2005.

[195] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

[196] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. In Y. Eldar and G. Kutyniok, editors, *Compressed Sensing*, chapter 5, pages 210–268. Cambridge University Press, 2012.

[197] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions of Information Theory*, 51(11):3697–3717, 2005.

[198] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

[199] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[200] H. Wang and A. Banerjee. Online alternating direction method. In *International Conference on Machine Learning (ICML)*, 2012.

[201] H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. *Neural Information Processing Systems (NIPS)*, 2014.

[202] H. Wang and A. Banerjee. Online randomized block coordinate descent. *ArXiv*, 2014.

[203] H. Wang, A. Banerjee, C. Hsieh, P. Ravikumar, and I. Dhillon. Large scale distributed sparse precesion estimation. In *Neural Information Processing Systems (NIPS)*, 2013.

[204] H. Wang, A. Banerjee, and Z. Luo. Parallel direction method of multipliers. In *Neural Information Processing Systems (NIPS)*, 2014.

[205] H. Wang, A. Banerjee, and Z. Luo. Parallel direction method of multipliers. *ArXiv*, 2014.

[206] H. Wang, F. Fazayeli, S. Chatterjee, and A. Banerjee. Gaussian copula precision estimation with missing values. 2014.

[207] X. Wang, M. Hong, S. Ma, and Z. Luo. Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers. *Preprint*, 2013.

[208] J. Wright and Y. Ma. Dense error correction via l1-minimization. *IEEE Transactions of Information Theory*, 56(7):3540–3560, 2010.

[209] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(2):210–227, February 2009.

[210] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research (JMLR)*, 11:2543–2596, 2010.

[211] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *arXiv*, 2014.

[212] L. Xue and H. Zou. Regularized rank-based estimation of high-dimensional nonparanormal graphical models. *The Annals of Statistics*, 40(5):2541–2571, 2012.

[213] A. Yang, S. Sastry, A. Ganesh, and Y. Ma. Fast l1-minimization algorithms and an application in robust face recognition: A review. In *International Conference on Image Processing (ICIP)*, pages 1849–1852, 2010.

[214] J. Yang and Y. Zhang. Alternating direction algorithms for L1-problems in compressive sensing. *ArXiv*, 2009.

[215] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation: an empirical study. *Journal of Machine Learning Research*, 7:1887–1907, 2006.

[216] Y. Yu. Better approximation and faster algorithm using the proximal average. In *Neural Information Processing Systems (NIPS)*, 2012.

[217] M. Yuan. High dimensional inverse covariance matrix estimation via linear programming. *Journal of Machine Learning Research*, 11, 2010.

[218] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society B*, 68:4967, 2007.

[219] X. M. Yuan. Alternating direction methods for sparse covariance selection. *Preprint*, 2009.

[220] X. M. Yuan and J. F. Yang. Sparse and low-rank matrix decomposition via alternating direction methods. *Preprint*, 2009.

[221] L. Zhang, M. Mahdavi, and R. Jin. Linear convergence with condition number independent access of full gradients. In *Neural Information Processing Systems (NIPS)*, 2013.

[222] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37:34683497, 2009.

[223] Z. Zhou, X. Li, J. Wright, E. Candes, and Y. Ma. Stable principal component pursuit. In *IEEE International Symposium on Information Theory*, 2010.

[224] D. Zimmerman, B. Zumbo, and R. Williams. Bias in estimation and hypothesis testing of correlation. *Transformation*, 24:133–158, 2003.

[225] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, pages 928–936, 2003.

[226] M. Zinkevich, M. Weimer, A. Smola, and L. Li. Parallelized stochastic gradient descent. In *Neural Information Processing Systems (NIPS)*, 2010.