Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

Junxiang Wang Date

The Applications of Alternating Minimization Algorithms on Deep Learning Models

By

Junxiang Wang
Doctor of Philosophy

Computer Science and Informatics

Liang Zhao, Ph.D.
Advisor

Carl Yang, Ph.D.
Committee Member

Li Xiong, Ph.D.
Committee Member

Lars Ruthotto, Ph.D.
Committee Member

Accepted:

Kimberly Jacob Arriola, Ph.D., MPH
Dean of the James T. Laney School of Graduate Studies

Date

The Applications of Alternating Minimization Algorithms on Deep Learning Models

By

Junxiang Wang
B.S., East China Normal University, Shanghai,China, 2012
M.Sc., George Mason University, VA, 2020

Advisor: Liang Zhao, Ph.D.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2022

Abstract

The Applications of Alternating Minimization Algorithms on Deep Learning Models
By Junxiang Wang

Gradient Descent(GD) and its variants are the most popular optimizers for training deep learning models. However, they suffer from many challenges such as gradient vanishing and backward locking, which prevent their more widespread use. To address these intrinsic drawbacks, Alternating Minimization(AM) methods have attracted attention from researchers as a potential way to train deep learning models. Their idea is to decompose a neural network into a series of linear and nonlinear equality constraints, which generate multiple subproblems and they can be minimized alternately. Their empirical evaluations demonstrate good scalability and high accuracy. They also avoid gradient vanishing problems and allow for non-differentiable activation functions, as well as allowing for complex non-smooth regularization and the constraints that are increasingly important for neural network architectures.

This dissertation discusses the applications of AM methods on deep learning models, which can be categorized into three research topics: 1). AM methods on Multi-Layer Perceptron(MLP), which includes deep learning Alternating Direction Method of Multipliers(dlADMM), and monotonous Deep Learning Alternating Minimization(mDLAM). 2). AM methods on Graph Neural Network(GNN), which includes the Invertible Validity-aware Graph Diffusion(IVGD). 3). AM methods for distributed neural network training, which includes parallel deep learning Alternating Direction Method of Multipliers(pdADMM), pdADMM-G, and pdADMM-G-Q.

For the dlADMM algorithm, parameters in each layer are updated in a backward and forward fashion. The time complexity is reduced from cubic to quadratic in(latent) feature dimensions for subproblems by iterative quadratic approximations and backtracking. Finally, we provide the convergence guarantee of the dlADMM algorithm under mild conditions with a sublinear convergence rate $o(1/k)$.

For the mDLAM algorithm, our innovative inequality-constrained formulation infinitely approximates the original problem with non-convex equality constraints, enabling our convergence proof of the proposed mDLAM algorithm regardless of the choice of hyperparameters. Our mDLAM algorithm is shown to achieve a fast linear convergence by the Nesterov acceleration technique.

For the IVGD model, we unroll AM methods into GNN models for graph source localization. Specifically, first, to inversely infer sources of graph diffusion, we propose a graph residual scenario to make existing graph diffusion models invertible with theoretical guarantees; second, we develop a novel error compensation mechanism that learns to offset the errors of the inferred sources. Finally, to ensure the validity of the inferred sources, we unroll AM methods into the proposed validity-aware layers to project inferred sources to feasible regions by encoding validity constraints. A linearization technique is proposed to strengthen the efficiency of our proposed layers. The convergence of the proposed IVGD is proven theoretically.

For the pdADMM algorithm, we achieve model parallelism by breaking layer dependency: parameters in each layer of neural networks can be updated independently

in parallel. The convergence of the proposed pdADMM to a stationary point is theoretically proven under mild conditions with a sublinear convergence rate $o(1/k)$.

For the pdADMM-G algorithm and the pdADMM-G-Q algorithm, in order to achieve model parallelism, we extend the proposed pdADMM algorithm to train the GA-MLP model, named the pdADMM-G algorithm. The extended pdADMM-G-Q algorithm reduces communication costs by introducing the quantization technique. Theoretical convergence to a (quantized) stationary point of two proposed algorithms is provided with a sublinear convergence rate $o(1/k)$.

The Applications of Alternating Minimization Algorithms on Deep Learning Models

By

Junxiang Wang
B.S., East China Normal University, Shanghai,China, 2012
M.Sc., George Mason University, VA, 2020

Advisor: Liang Zhao, Ph.D.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Computer Science and Informatics
2022

Acknowledgments

Firstly, I would like to express my heartfelt gratitude to to my mentor, Dr. Liang Zhao, for his support during my Ph.D study. His effective instruction and mental encouragement helped me overcome the difficulty of my research projects. I really appreciate the efforts Dr Liang Zhao made when he revised my research papers and presentation slides over and over. Without his guidance, I would never be able to publish many papers during these years.

Secondly, I appreciate all my committee members, Dr.Carl Yang, Dr. Li Xiong, and Dr. Lars Ruthotto. They provided insightful comments to the my preliminary dissertation prospectus, and the draft of the dissertation thesis. Their useful suggestions broadened my research horizon, and left me equipped with ideas of new research directions. Special thanks go to Dr.Yue Cheng, who provided valuable suggestions and hardware support from the level of system infrastructures during our research collaboration.

Thirdly, I would like to thank my research collaborators in and out of the Dr. Liang Zhao's laboratory: Hongyi Li, Xiaojie Guo, Chen Ling, Junji Jiang, Shiyu Wang, Guangji Bai, Yuyang Gao, and Zheng Chai, to name but a few. Thank you all for your unique contributions to our papers and useful advices to improve our research ideas.

Finally, I would like to dedicate this dissertation thesis to my girlfriend, my parents, and other family members. They provided me with support and courage during this memorable journey.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Deep learning has been the focus of the machine learning community during the last decade. While conventional machine learning models such as Support Vector Machine(SVM) require some assumptions(e.g., the margin should be as large as possible), and they have limited capacity to process natural data in their raw forms, deep learning models have flexible forms and entail no requirement, and they are composed of non-linear modules that can learn multiple levels of representations automatically [59]. Because deep learning models perform surprisingly outstandingly on large-scale datasets, they require efficient optimizers to obtain a feasible solution within realistic time limits.

Gradient Descent(GD) was the most popular optimizer for training deep learning models one decade ago due to its simplicity and effectiveness. However, it has some drawbacks such as painful hyper-parameter tuning, and the possibility to plunge into saddle points [27]. To address these issues, many variants of GD approaches have been introduced and are well-known, including but not limited to GD momentum [91], AdaGrad [28], AdaDelta [118] and Adam [52]. Among them, GD momentum, AdaGrad, and AdaDelta improve GD via the first moment of the gradient, whereas Adam improves them further via the second moment of the gradient. However, they

suffer from many challenges that prevent their more widespread use: the gradient vanishing is one major challenge, which means the error signal diminishes as the gradient is backpropagated. For example, a deep Recurrent Neural Network(RNN) model easily suffers from the gradient vanishing problem because activation functions are sigmoid or tanh, and their derivatives are smaller than 1 [76]. Poor conditioning is another challenge, where a small change of input leads to a drastic change in the gradient. Even though researchers have proposed many solutions(e.g. Rectified Linear Unit(ReLU) [1], skip connection [37] and batch normalization [45]), these problems still exist because the nested structures of deep learning models amplify the effects of existing challenges such as gradient vanishing and poor conditioning. For example, a narrow Multi-Layer Perceptron(MLP) model with skip connections and ReLU may still be prone to the gradient vanishing problem.

To tackle these intrinsic drawbacks of gradient descent optimization methods, Alternating Minimization(AM) methods have started to attract attention as a potential way to solve deep learning problems. A neural network problem is reformulated as a nested function associated with multiple linear and nonlinear transformations across multi-layers. This nested structure is then decomposed into a series of linear and nonlinear equality constraints by introducing auxiliary variables and penalty hyperparameters. The linear and nonlinear equality constraints generate multiple subproblems, which can be minimized alternately. Many recent AM methods have focused on applying the Alternating Direction Method of Multipliers(ADMM) [93, 101], Block Coordinate Descent(BCD) [120] and Method of Auxiliary Coordinates(MAC) [9] to replace a nested neural network with a constrained problem without nesting, with empirical evaluations demonstrating good scalability in terms of the number of layers and high accuracy on the test sets. These methods also avoid gradient vanishing problems and allow for non-differentiable activation functions such as binarized neural networks [22], as well as allowing for complex non-smooth regularization and the

Table 1.1: Notations of the MLP training problem in this dissertation.

Notations	Descriptions
L	The number of layers.
W_l	The weight matrix for the l-th layer.
z_l	The output of the linear mapping for the l-th layer.
$f_l(z_l)$	The nonlinear activation function for the l-th layer.
a_l	The output for the l-th layer.
x	The input matrix of the neural network.
y	The predefined label vector.
$R(z_L, y)$	The risk function.
n_l	The number of neurons for the l-th layer.

constraints that are increasingly important for deep neural architectures that are required to satisfy practical requirements such as interpretability, energy-efficiency, and cost awareness [9]. The ADMM, as a representative of AM methods, has been explored extensively for different neural network architectures. It was first used to solve the MLP problem [93], and then was extended to other architectures such as RNN [92].

This dissertation studies how AM methods can be utilized to train the MLP model, which is the most fundamental model in deep learning. Table 1.1 lists important notations utilized in this dissertation. A typical MLP model is defined by multiple linear mappings and nonlinear activation functions. A linear mapping for the l-th layer is composed of a weight matrix $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$, where n_l is the number of neurons for the l-th layer; a nonlinear mapping for the l-th layer is defined by a continuous activation function $f_l(\bullet)$. Given an input $a_{l-1} \in \mathbb{R}^{n_{l-1}}$ from the $(l-1)$-th layer, the l-th layer outputs $a_l = f_l(W_l a_{l-1})$. Obviously, a_{l-1} is nested in $a_l = f_l(\bullet)$. By introducing z_l which serves as the output of the linear mapping for the l-th layer, the task of training the MLP model is formulated mathematically as follows:

Problem 1.

$$\min_{W_l, z_l, a_l} R(z_L, y) \; s.t. \; z_l = W_l a_{l-1} (l = 1, \cdots, L), \; a_l = f_l(z_l) \; (l = 1, \cdots, L-1)$$

In Problem 1, $a_0 = x \in \mathbb{R}^{n_0}$ is the input of the MLP model where n_0 is the number of feature dimensions, and y is a predefined label vector. $R(z_L, y)$ is a risk function for the L-th layer, which is convex, continuous, and proper.

The goal of this dissertation is to develop optimization algorithms based on AM methods to address Problem 1. While some existing papers such as [9, 93, 120] studied this problem, they suffer from several challenges, which will be addressed in this dissertation: **1. The lack of theoretical guarantees.** While many previous papers have studied the convergence theory of AM methods on nonconvex problems, they cannot be directly applied to neural networks. This is because a typical neural network model is a nested formulation of nonlinear functions, which causes the objective to be highly non-convex. **2. Expensive computations of subproblems.** Some subproblems generated by AM methods are computationally expensive to address. This is because they involve matrix inversion. Computing an inverse matrix needs further sub-iterations, and its time complexity is approximate $O(n^3)$, where n is a feature dimension [6]. **3. Slow convergence toward solutions.** For some AM methods such as ADMM, their empirical convergence curves are long-tailed. In other words, they usually take hundreds of iterations to converge to high accuracy, even for simple examples [6]. It is often the case that they are stuck on a modest solution. **4. The lack of investigations to parallel training neural networks.** Existing state-of-the-art algorithms are still based on GD to achieve parallel training of neural networks. However, they are subject to the backward locking problem [44]: the gradient calculations in one layer tightly depend on and have to wait for results of all previous layers, which prevents gradients of different layers from being calculated in parallel. While AM methods have great potential to address this drawback, they are rarely explored, developed, and evaluated.

In order to address all these challenges, we propose several AM methods to handle these challenges in the dissertation. They can be categorized into three research

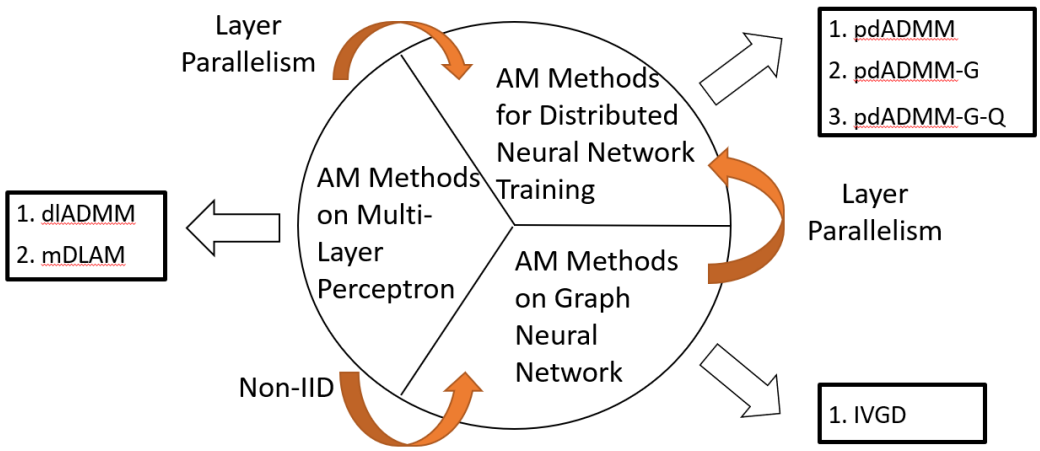


Figure 1.1: The relationships among three research topics.

topics illustrated by Figure 1.1, all of which are shown below:

1. AM methods on Multi-Layer Perceptron(MLP). Two AM methods, the deep learning Alternating Direction Method of Multipliers(dlADMM) [101] and monotonous Deep Learning Alternating Minimization(mDLAM) [106], are proposed to train MLP models. They will be introduced in Chapters 2 and 3, respectively.

In the dlADMM algorithm, parameters of a neural network are updated backward and then forward so that the parameter information is exchanged efficiently(i.e. address Challenge 3); the time complexity for subproblems is reduced from cubic to quadratic in(latent) feature dimensions via a dedicated algorithm that utilizes iterative quadratic approximations and backtracking(i.e. address Challenge 2). Finally, we provide the first proof of global convergence for an ADMM-based method(dlADMM) in a deep neural network problem under mild conditions(i.e. address challenge 1).

The mDLAM algorithm approximates the original problem by an innovative formulation with inequality constraints, enabling our convergence proof of the proposed mDLAM algorithm regardless of the choice of hyperparameters(i.e. address Challenge 1). It is shown to achieve a fast linear convergence by the Nesterov acceleration technique(i.e. address Challenges 1 and 3).

2. AM methods on Graph Neural Network(GNN). We extend the applications of AM methods from the MLP to the GNN because GNN models deal with non-IID data (i.e. nodes and their connections in a graph), and the MLP can be considered as a special case of GNN models. GNN models incorporate node attributes into models and learn node representations effectively by capturing network topology and neighboring information [53]. Moreover, they can "learn" rules from the data in an end-to-end fashion. We devise GNN models via AM methods to fulfill graph learning tasks. For example, graph source localization aims to detect source nodes of a graph given their future graph diffusion patterns. It covers a wide range of real-world applications such as misinformation detection [46], Email virus localization [75], and malware detection [49].

We propose a novel GNN-based model, Invertible Validity-aware Graph Diffusion(IVGD), for graph source localization via unrolling AM methods into neural networks [104]. Specifically, first, to inversely infer sources of graph diffusion, we propose a graph residual scenario to make existing graph diffusion models invertible with theoretical guarantees; second, we develop a novel error compensation mechanism that learns to offset the errors of the inferred sources. Finally, to ensure the validity of the inferred sources, we unroll AM methods into the proposed validity-aware layers to project inferred sources to feasible regions by encoding validity constraints. A linearization technique is proposed to strengthen the efficiency of our proposed layers(i.e. address Challenge 2). The convergence of the proposed IVGD is proven theoretically(i.e. address Challenge 1). Extensive experiments on nine real-world datasets demonstrate that our proposed IVGD outperforms state-of-the-art comparison methods significantly.

3. AM methods for distributed neural network training. We extend AM methods on MLP and GNN models to realize distributed training via model parallelism (i.e. layerwise parallelism). The proposed AM methods include parallel deep

learning Alternating Direction Method of Multipliers(pdADMM) [103], the graph pdADMM(pdADMM-G) [105], and the quantized graph pdADMM(pdADMM-G-Q) [105]. They will be introduced in Chapters 4 and 5, respectively.

The pdADMM algorithm achieves layer parallelism via breaking layer dependency among variables so that parameters in each layer of neural networks can be updated independently in parallel(i.e. address Challenge 4). The convergence of the proposed pdADMM to a stationary point is theoretically proven under mild conditions(i.e. address Challenge 1).

We extend the pdADMM algorithm to train Graph Neural Networks(GNNs) such as the Graph-Augmented Multi-Layer Perceptron(GA-MLP) model, which leads to the pdADMM-G algorithm, and the pdADMM-G-Q algorithm extends the pdADMM-G algorithm by the quantization technique, in order to reduce communication overheads(i.e. address Challenge 4). Theoretical convergence to a (quantized) stationary point of the pdADMM-G algorithm and the pdADMM-G-Q algorithm is provided with a sublinear convergence rate $o(1/k)$, where k is the number of iterations(i.e. address Challenge 1).

1.1 Contributions

The major proposed research contributions can be stated as follows.

The dlADMM Algorithm:

- **We present a novel and efficient dlADMM algorithm to handle the MLP learning problem.** The new dlADMM updates parameters in a backward and then forward fashion to speed up the convergence process.

- **We propose the use of quadratic approximation and backtracking techniques to avoid the need for matrix inversion as well as reduce the computational cost for large-scale datasets**. The time complexity of

subproblems in dlADMM is reduced from $O(n^3)$ to $O(n^2)$.

- **We investigate several attractive convergence properties of dlADMM.** The convergence assumptions are very mild to ensure that most deep learning applications satisfy our assumptions. dlADMM is guaranteed to converge to a stationary point globally(i.e., whatever the initialization is) when the hyper-parameter is sufficiently large. We also analyze the new algorithm's sublinear convergence rate.

- **We conduct experiments on several benchmark datasets to validate our proposed dlADMM algorithm.** The experimental results show that the proposed dlADMM algorithm performs better than most existing state-of-the-art algorithms, including GD and its variants.

The mDLAM Algorithm:

- **We propose a novel formulation for neural network optimization.** The deeply nested activation functions are disentangled into separate functions innovatively coordinated by inherently convex inequality constraints.

- **We present an accelerated optimization algorithm.** A quadratic approximation technique is utilized to avoid matrix inversion. Every subproblem has a closed-form solution. The Nesterov acceleration technique is applied to further boost convergence.

- **We investigate the convergence of the proposed mDLAM algorithm under mild conditions.** The new mDLAM algorithm is guaranteed to converge to a stationary point whatever hyperparameters we choose. Furthermore, the proposed mDLAM algorithm is shown to achieve a linear convergence rate, which is faster than existing methods.

- **Extensive experiments are conducted to demonstrate the effectiveness of the proposed mDLAM algorithm.** We test our proposed mDLAM algorithm on four benchmark datasets. Experimental results illustrate that our proposed mDLAM algorithm is linearly convergent on four datasets, and outperforms consistently state-of-the-art optimizers. Sensitivity analysis of the running time shows that it increases linearly with the increase of neurons and hyperparameters.

The IVGD Model:

- **Design a generic end-to-end framework for source location.** We develop a framework for the inverse of graph diffusion models, and learn the rules of graph diffusion models automatically. It does not require hand-crafted rules and can be used for source localization. Our framework is generic to any graph diffusion model, and the code has been released publicly.

- **Develop an invertible graph diffusion model with an error compensation mechanism.** We propose a new graph residual net with Lipschitz regularization to ensure the invertibility of graph diffusion models. Furthermore, we propose an error compensation mechanism to offset the errors inferred from the graph residual net.

- **Propose an efficient validity-aware layer to maintain the validity of inferred sources.** Our model can ensure the validity of inferred sources by automatically learning validity-aware layers. We further accelerate the optimization of the proposed layers by leveraging a linearization technique. It transforms nonconvex problems into convex problems, which have closed-form solutions. Moreover, we provide the convergence guarantees of the proposed IVGD to a feasible solution.

- **Conduct extensive experiments on nine datasets.** Extensive experiments on nine datasets have been conducted to demonstrate the effectiveness and robustness of our proposed IVGD. Our proposed IVGD outperforms all comparison methods significantly on five metrics, especially 20% on F1-Score.

The pdADMM Algorithm:

- **We propose a novel reformulation of the MLP training problem.** The formulation splits a neural network into independent layer partitions and allows ADMM to achieve model parallelism.

- **We present a model-parallelism version of the ADMM algorithm to train an MLP**. All parameters in each layer can be updated in parallel to speed up the training process significantly. All subproblems generated by the pdADMM algorithm are discussed in detail.

- **We investigate the convergence properties of parallel ADMM**. For the common nonlinear activation functions such as the Rectified linear unit(ReLU), we prove that the pdADMM converges to a state-of-the-art stationary point with a sublinear convergence rate $o(1/k)$.

- **We conduct extensive experiments on six benchmark datasets.** Experimental results show the massive speedup of the proposed pdADMM as well as its competitive performance with state-of-the-art optimizers.

The pdADMM-G and pdADMM-G-Q Algorithms:

- **We propose a novel reformulation of GA-MLP models**. It splits a neural network into independent layer partitions and allows ADMM to achieve model parallelism.

- **We propose a novel pdADMM-G framework to train a GA-MLP model**. All subproblems generated by the ADMM algorithm are discussed.

The extended pdADMM-G-Q algorithm reduces communication costs by introducing the quantization technique.

- **We provide the theoretical convergence guarantee of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm**. Specifically, they converge to a(quantized) stationary point of GA-MLP models when the hyperparameters are sufficiently large, and their sublinear convergence rates are $o(1/k)$.

- **We conduct extensive experiments to show the effectiveness of two proposed algorithms**. Experiments on nine benchmark datasets show the convergence, the massive speedup of the proposed pdADMM-G algorithm and the proposed pdADMM-G-Q algorithm, as well as their outstanding performance when compared with all state-of-the-art optimizers. Moreover, the proposed pdADMM-G-Q algorithm reduces communication overheads by up to 45%.

1.2 The Organization of the dissertation

The remaining dissertation is organized as follows: Chapter 2 proposes the dlADMM algorithm. Chapter 3 presents the mDLAM algorithm. The pdADMM algorithm is introduced in Chapter 4. The pdADMM-G algorithm and the pdADMM-G-Q algorithm are proposed in Chapter 5. Chapter 6 summarizes this dissertation and discusses future work.

Chapter 2

The dlADMM Algorithm

2.1 Introduction

As introduced in Chapter 1, Gradient Descent(GD) and its variants are state-of-the-art optimizers for training deep learning models. But they suffer from many limitations such as gradient vanishing and poor conditioning. Recently, the use of the Alternating Direction Method of Multipliers(ADMM) has been proposed as an alternative to GD. The ADMM splits a problem into many subproblems and coordinates them globally to obtain the solution. It has been demonstrated successfully for many machine learning applications [6]. The advantages of ADMM are numerous: it exhibits linear scaling as data is processed in parallel across cores; it does not require gradient steps and hence avoids gradient vanishing problems; it is also immune to poor conditioning [93].

Even though the performance of the ADMM seems promising, there are still several challenges at must be overcome: **1. The lack of global convergence guarantees.** Even though many empirical experiments have shown that ADMM converges in deep learning applications, the underlying theory governing this convergence behavior remains mysterious. This is because a typical deep learning problem consists

of a combination of linear and nonlinear mappings, causing optimization problems to be highly nonconvex. This means that traditional proof techniques cannot be directly applied. **2. Slow convergence towards solutions.** Although ADMM is a powerful optimization framework that can be applied to large-scale deep learning applications, it usually converges slowly to high accuracy, even for simple examples [6]. It is often the case that ADMM becomes trapped in a modest solution and hence performs worse than GD, as the experiment described later in this chapter in Section 2.5 demonstrates. **3. Cubic time complexity concerning feature dimensions.** The implementation of the ADMM is very time-consuming for real-world datasets. Experiments conducted by Taylor et al. found that ADMM required more than 7000 cores to train a neural network with just 300 neurons [93]. This computational bottleneck mainly originates from the matrix inversion required to update the weight parameters. Computing an inverse matrix needs further sub-iterations, and its time complexity is approximate $O(n^3)$, where n is a feature dimension [6].

To deal with these difficulties simultaneously, in this chapter we propose a novel optimization framework for a deep learning Alternating Direction Method of Multipliers(dlADMM) algorithm. Specifically, our new dlADMM algorithm updates parameters first in a backward direction and then forwards. This update approach propagates parameter information across the whole network and accelerates the convergence process. It also avoids the operation of matrix inversion using the quadratic approximation and backtracking techniques, reducing the time complexity from $O(n^3)$ to $O(n^2)$. Finally, to the best of our knowledge, we provide the first proof of the global convergence of the ADMM-based method(dlADMM) in a deep neural network problem. The assumption conditions are mild enough for many common loss functions(e.g. cross-entropy loss and square loss) and activation functions(e.g. Rectified Linear Unit(ReLU) and leaky ReLU) to satisfy. Our proposed framework and convergence proof are highly flexible for Multi-Layer Perceptron(MLP) neural networks, as

well as being easily extendable to other popular network architectures such as Convolutional Neural Networks [54] and Recurrent Neural Networks [69]. Our contributions in this chapter include:

- We present a novel and efficient dlADMM algorithm to handle the MLP training problem. The new dlADMM updates parameters in a backward-forward fashion to speed up the convergence process.

- We propose the use of quadratic approximation and backtracking techniques to avoid the need for matrix inversion as well as to reduce the computational cost for large-scale datasets. The time complexity of subproblems in dlADMM is reduced from $O(n^3)$ to $O(n^2)$.

- We investigate several attractive convergence properties of dlADMM. The convergence assumptions are very mild to ensure that most deep learning applications satisfy our assumptions. dlADMM is guaranteed to converge to a stationary point globally(i.e., whatever the initialization is) when the hyperparameter is sufficiently large. We also analyze the new algorithm's sublinear convergence rate.

- We conduct experiments on several benchmark datasets to validate our proposed dlADMM algorithm. The results show that the proposed dlADMM algorithm performs better than most existing state-of-the-art algorithms, including GD and its variants.

The rest of this chapter is organized as follows. In Section 2.2, we summarize recent research related to this topic. In Section 2.3, we present the new dlADMM algorithm, quadratic approximation, and the backtracking techniques utilized. In Section 2.4, we introduce the main convergence results for the Proposed dlADMM algorithm. The results of extensive experiments conducted to show the convergence, efficiency, and

effectiveness of our proposed new dlADMM algorithm are presented in Section 2.5, and Section 2.6 concludes this chapter by summarizing the research.

2.2 Related Work

Previous literature related to this research includes optimization for deep learning models and ADMM for nonconvex problems.

Optimization for Deep Learning Models: The GD algorithm and its variants play a dominant role in the research conducted by the deep learning optimization community. The famous back-propagation algorithm was firstly introduced by Rumelhart et al. to train the neural network effectively [85]. Since the superior performance exhibited by AlexNet [54] in 2012, deep learning has attracted a great deal of researchers' attention and many new optimizers based on GD have been proposed to accelerate the convergence process, including the use of Polyak momentum [80], as well as research on the Nesterov momentum and initialization by Sutskever et al. [91]. Adam is the most popular method because it is computationally efficient and requires little tuning [52]. Other well-known methods that incorporate adaptive learning rates include AdaGrad [28], RMSProp [95], and AMSGrad [82]. Recently, the Alternating Direction Method of Multipliers(ADMM) has become popular with researchers due to its excellent scalability [93]. However, even though these optimizers perform well in real-world applications, their convergence mechanisms remain mysterious. This is because convergence assumptions do not apply to deep learning problems, which often require non-differentiable activation functions such as ReLU.

Convergence Analysis of Nonconvex ADMM: Despite the outstanding performance of the nonconvex ADMM, its convergence theory is not well established due to the complexity of both coupled objectives and various inequality and equality constraints. Specifically, Magnusson et al. provided new convergence conditions of

ADMM for a class of nonconvex structured optimization problems [67]; Li and Pong investigated the properties of the nonconvex ADMM on the composite optimization problem [60]; Wang et al. presented mild convergence conditions of the nonconvex ADMM where the objective function can be coupled and nonsmooth [107]; Hong et al. proved that the classic ADMM converges to stationary points provided that the penalty parameter is sufficiently large [40]; Wang et al. proved the convergence of the multi-convex ADMM [100]; Liu et al. proved the convergence properties of a parallel and linearized ADMM [64]; Xie et al. proposed a deep-learning-based ADMM algorithm to study the constrained optimization problems [113]. For more work, please refer to [14, 33, 34, 94, 98, 102].

2.3 dlADMM

We present our proposed dlADMM algorithm in this section. Section 2.3.1 formulates the deep neural network problem, Section 2.3.2 introduces how the Proposed dlADMM algorithm works, and the quadratic approximation and backtracking techniques used to solve the subproblems are presented in Section 2.3.3.

2.3.1 Problem Transformation

Problem 1 is difficult to solve because of the nonlinear constraint $a_l = f_l(z_l)$. To address this, we can relax Problem 1 by adding an ℓ_2 penalty to address Problem 2 as follows:

Problem 2.

$$\min_{W_l,z_l,a_l} F(\mathbf{W},\mathbf{z},\mathbf{a}) = R(z_L,y) + (\nu/2)\sum_{l=1}^{L-1}(\|z_l - W_l a_{l-1}\|_2^2 + \|a_l - f_l(z_l)\|_2^2)$$
$$s.t.\ z_L = W_L a_{L-1}$$

where $\mathbf{W} = \{W_l\}_{l=1}^{L}$, $\mathbf{z} = \{z_l\}_{l=1}^{L}$, $\mathbf{a} = \{a_l\}_{l=1}^{L-1}$ and $\nu > 0$ is a tuning parameter. Compared with Problem 1, Problem 2 has only a linear constraint $z_L = W_L a_{L-1}$ and hence is easier to solve. It is straightforward to show that as $\nu \to \infty$, the solution to Problem 2 approaches that of Problem 1.

2.3.2 the Proposed dlADMM Algorithm

We introduce the Proposed dlADMM algorithm to solve Problem 2 in this section. The traditional ADMM strategy for optimizing parameters is to start from the first layer and then update parameters in the following layer sequentially [93]. In this case, the parameters in the final layer are subject to the parameter update in the first layer. However, the parameters in the final layer contain important information that can be transmitted to the previous layers to speed up convergence. To achieve this, we propose our novel dlADMM framework, as shown in Figure 2.1. Specifically, the Proposed dlADMM algorithm updates parameters in two steps. In the first, the Proposed dlADMM begins updating from the L-th(final) layer and moves backward toward the first layer. The update order of parameters in the same layer is $a_l \to z_l \to W_l$. In the second, the Proposed dlADMM reverses the update direction, beginning at the first layer and moving forward toward the L-th(final) layer. The update order of the parameters in the same layer is $W_l \to z_l \to a_l$. The parameter information for all layers can be exchanged completely by adopting this update approach.

Now we can present our dlADMM algorithm mathematically. The augmented Lagrangian function of Problem 2 is shown in the following form:

$$L_\rho(\mathbf{W}, \mathbf{z}, \mathbf{a}, u) = R(z_L, y) + \phi(\mathbf{W}, \mathbf{z}, \mathbf{a}, u) \tag{2.1}$$

where $\phi(\mathbf{W}, \mathbf{z}, \mathbf{a}, u) = (\nu/2)\sum_{l=1}^{L-1}(\|z_l - W_l a_{l-1}\|_2^2 + \|a_l - f_l(z_l)\|_2^2) + u^T(z_L - W_L a_{L-1}) + (\rho/2)\|z_L - W_L a_{L-1}\|_2^2$, u is a dual variable and $\rho > 0$ is a hyperparameter of the

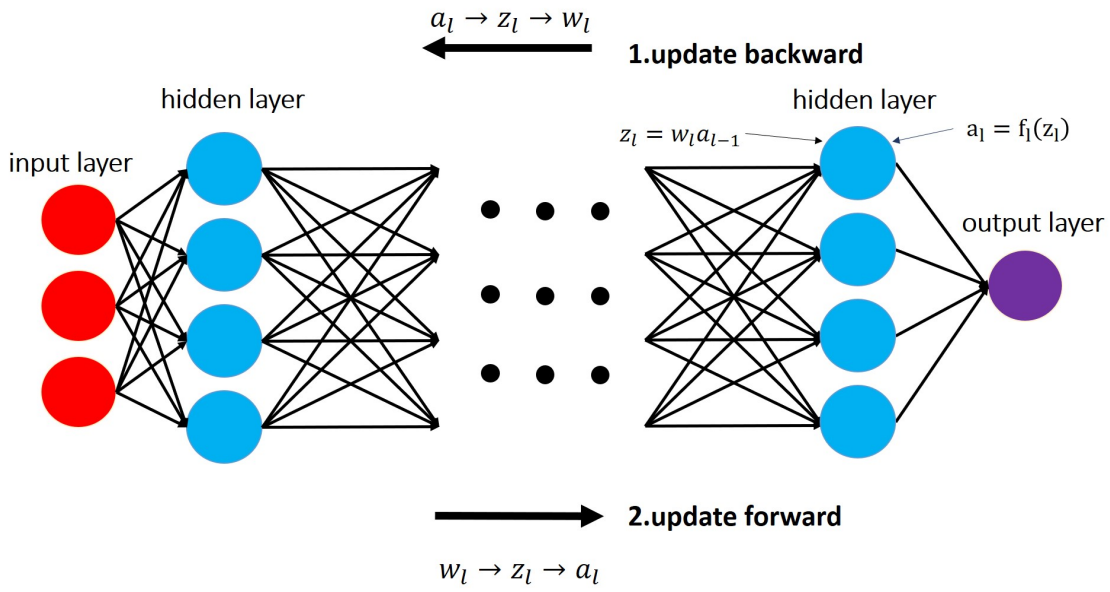


Figure 2.1: The overview of the proposed dlADMM algorithm.

Proposed dlADMM algorithm. We denote $\overline{W}_l^{k+1}$, $\overline{z}_l^{k+1}$ and $\overline{a}_l^{k+1}$ as the backward update of the Proposed dlADMM for the l-th layer in the $(k+1)$-th iteration , while W_l^{k+1}, z_l^{k+1} and a_l^{k+1} are denoted as the forward update of the Proposed dlADMM for the l-th layer in the $(k+1)$-th iteration. Moreover, we denote $\overline{\mathbf{W}}_l^{k+1} = \{\{W_i^k\}_{i=1}^{l-1}, \{\overline{W}_i^{k+1}\}_{i=l}^{L}\}$, $\overline{\mathbf{z}}_l^{k+1} = \{\{z_i^k\}_{i=1}^{l-1}, \{\overline{z}_i^{k+1}\}_{i=l}^{L}\}$, $\overline{\mathbf{a}}_l^{k+1} = \{\{a_i^k\}_{i=1}^{l-1}, \{\overline{a}_i^{k+1}\}_{i=l}^{L-1}\}$, $\mathbf{W}_l^{k+1} = \{\{W_i^{k+1}\}_{i=1}^{l}, \{\overline{W}_i^{k+1}\}_{i=l+1}^{L}\}$, $\mathbf{z}_l^{k+1} = \{\{z_i^{k+1}\}_{i=1}^{l}, \{\overline{z}_i^{k+1}\}_{i=l+1}^{L}\}$, $\mathbf{a}_l^{k+1} = \{\{a_i^{k+1}\}_{i=1}^{l}, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}\}$, $\overline{\mathbf{W}}^{k+1} = \{\overline{W}_i^{k+1}\}_{i=1}^{L}$, $\overline{\mathbf{z}}^{k+1} = \{\overline{z}_i^{k+1}\}_{i=1}^{L}$, $\overline{\mathbf{a}}^{k+1} = \{\overline{a}_i^{k+1}\}_{i=1}^{L-1}$, $\mathbf{W}^{k+1} = \{W_i^{k+1}\}_{i=1}^{L}$, $\mathbf{z}^{k+1} = \{z_i^{k+1}\}_{i=1}^{L}$, and $\mathbf{a}^{k+1} = \{a_i^{k+1}\}_{i=1}^{L-1}$. Then the Proposed dlADMM algorithm is shown in Algorithm 1. Specifically, Lines 5, 6, 10, 13, 15, and 16 solve six subproblems, namely, $\overline{a}_l^{k+1}$, $\overline{z}_l^{k+1}$, $\overline{W}_l^{k+1}$, W_l^{k+1}, z_l^{k+1} and a_l^{k+1}, respectively. Lines 21 and 22 update the residual r^{k+1} and the dual variable u^{k+1}, respectively.

Algorithm 1: The proposed dlADMM algorithm

Require: y, $a_0 = x$, ρ, ν.
Ensure: $a_l(l = 1, \cdots, L-1), W_l(l = 1, \cdots, L), z_l(l = 1, \cdots, L)$.

Initialize $k = 0$.
while $\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}$ not converged **do**
  for $l = L$ to 1 **do**
    if $l < L$ **then**
      Update $\overline{a}_l^{k+1}$ in Equation (2.3).
      Update $\overline{z}_l^{k+1}$ in Equation (2.4).
    else
      Update $\overline{z}_L^{k+1}$ in Equation (2.5).
    end if
    Update $\overline{W}_l^{k+1}$ in Equation (2.7).
  end for
  for $l = 1$ to L **do**
    Update W_l^{k+1} in Equation (2.9).
    if $l < L$ **then**
      Update z_l^{k+1} in Equation (2.10).
      Update a_l^{k+1} in Equation (2.12).
    else
      Update z_L^{k+1} in Equation (2.11).
      $r^{k+1} \leftarrow z_L^{k+1} - W_L^{k+1} a_{L-1}^{k+1}$.
      $u^{k+1} \leftarrow u^k + \rho r^{k+1}$.
    end if
  end for
  $k \leftarrow k + 1$.
end while
Output $\mathbf{W}, \mathbf{z}, \mathbf{a}$.

2.3.3 The Quadratic Approximation and Backtracking

The six subproblems in Algorithm 1 are discussed in detail in this section. Most can be solved by quadratic approximation and the backtracking techniques described above, so the operation of the matrix inversion can be avoided.

1. Update $\overline{a}_l^{k+1}$

The variables $\overline{a}_l^{k+1}(l = 1, \cdots, L-1)$ are updated as follows:

$$\overline{a}_l^{k+1} \leftarrow \arg\min_{a_l} L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \{a_i^k\}_{i=1}^{l-1}, a_l, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k)$$

The subproblem is transformed into the following form after it is replaced by Equation (2.1).

$$\overline{a}_l^{k+1} \leftarrow \arg\min_{a_l} \phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \{a_i^k\}_{i=1}^{l-1}, a_l, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k) \tag{2.2}$$

Because a_l and W_{l+1} are coupled in $\phi(\bullet)$, in order to solve this problem, we must compute the inverse matrix of $\overline{W}_{l+1}^{k+1}$, which involves sub-iterations and is computationally expensive [93]. In order to handle this challenge, we define $\overline{Q}_l(a_l; \overline{\tau}_l^{k+1})$ as a quadratic approximation of ϕ at a_l^k, which is mathematically reformulated as follows:

$$\begin{aligned}\overline{Q}_l(a_l; \overline{\tau}_l^{k+1}) = \phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l+1}^{k+1}, u^k) + (\nabla_{a_l^k}\phi)^T(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l+1}^{k+1}, u^k)(a_l - a_l^k) \\ + \|\overline{\tau}_l^{k+1} \circ (a_l - a_l^k)^{\circ 2}\|_1/2\end{aligned}$$

where $\overline{\tau}_l^{k+1} > 0$ is a parameter vector, $\circ$ denotes the Hadamard product(i.e. the elementwise product), and $a^{\circ b}$ denotes a to the Hadamard power of b and $\| \bullet \|_1$ is the ℓ_1 norm. $\nabla_{a_l^k}\phi$ is the gradient of $\overline{a}_l$ at a_l^k. $\overline{Q}_l(a_l^k; \overline{\tau}_l^{k+1}) = \phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l+1}^{k+1}, u^k)$. Rather than minimizing the original problem in Equation (2.2), we instead solve the

following problem:

$$\overline{a}_l^{k+1} \leftarrow \arg\min_{a_l} \overline{Q}_l(a_l; \overline{\tau}_l^{k+1}) \tag{2.3}$$

Because $\overline{Q}_l(a_l; \overline{\tau}_l^{k+1})$ is a quadratic function with respect to a_l, the solution can be obtained by

$$\overline{a}_l^{k+1} \leftarrow a_l^k - \nabla_{a_l^k}\phi/\overline{\tau}_l^{k+1}$$

given a suitable $\overline{\tau}_l^{k+1}$. Now the main focus is how to choose $\overline{\tau}_l^{k+1}$. Algorithm 2 shows the backtracking algorithm utilized to find a suitable $\overline{\tau}_l^{k+1}$. Lines 2-5 implement a while loop until the condition $\phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_l^{k+1}, u^k) \leq \overline{Q}_l(\overline{a}_l^{k+1}; \overline{\tau}_l^{k+1})$ is satisfied. As $\overline{\tau}_l^{k+1}$ becomes larger and larger, $\overline{a}_l^{k+1}$ is close to a_l^k and a_l^k satisfies the loop condition, which precludes the possibility of the infinite loop. The time complexity of Algorithm 2 is $O(n^2)$, where n is the number of features or neurons.

Algorithm 2: The Backtracking Algorithm to update $\overline{a}_l^{k+1}$

Require: $\overline{\mathbf{W}}_{l+1}^{k+1}$, $\overline{\mathbf{z}}_{l+1}^{k+1}$, $\overline{\mathbf{a}}_{l+1}^{k+1}$, u^k, ρ, some constant $\overline{\eta} > 1$.
Ensure: $\overline{\tau}_l^{k+1}$,$\overline{a}_l^{k+1}$.
1: Pick up $\overline{t}$ and $\overline{\beta} = a_l^k - \nabla_{a_l^k}\phi/\overline{t}$
2: **while** $\phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \{a_i^k\}_{i=1}^{l-1}, \overline{\beta}, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k) > \overline{Q}_l(\overline{\beta}; \overline{t})$ **do**
3: $\quad \overline{t} \leftarrow \overline{t}\overline{\eta}$.
4: $\quad \overline{\beta} \leftarrow a_l^k - \nabla_{a_l^k}\phi/\overline{t}$.
5: **end while**
6: Output $\overline{\tau}_l^{k+1} \leftarrow \overline{t}$.
7: Output $\overline{a}_l^{k+1} \leftarrow \overline{\beta}$.

2. Update $\overline{z}_l^{k+1}$

The variables $\overline{z}_l^{k+1}(l = 1, \cdots, L)$ are updated as follows:

$$\overline{z}_l^{k+1} \leftarrow \arg\min_{z_l} L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \{z_i^k\}_{i=1}^{l-1}, z_l, \{\overline{z}_i^{k+1}\}_{i=l+1}^{L}, \overline{\mathbf{a}}_l^{k+1}, u^k)$$

which is equivalent to the following forms: for $\overline{z}_l^{k+1}(l = 1, \cdots, L-1)$,

$$\overline{z}_l^{k+1} \leftarrow \arg\min_{z_l} \phi(\overline{\mathbf{W}}_{l+1}^{k+1}, \{z_i^k\}_{i=1}^{l-1}, z_l, \{\overline{z}_i^{k+1}\}_{i=l+1}^{L}, \overline{\mathbf{a}}_l^{k+1}, u^k) \tag{2.4}$$

and for $\overline{z}_L^{k+1}$,

$$\overline{z}_L^{k+1} \leftarrow \arg\min_{z_L} \phi(\mathbf{W}^k, \{z_i^k\}_{i=1}^{L-1}, z_L, \mathbf{a}^k, u^k) + R(z_L, y) \tag{2.5}$$

Equation (2.4) is highly nonconvex because the nonlinear activation function $f(z_l)$ is contained in $\phi(\bullet)$. For common activation functions such as ReLU and leaky ReLU, Equation (2.4) has a closed-form solution; for other activation functions like sigmoid and hyperbolic tangent(tanh), a look-up table is recommended [93].

Equation (2.5) is a convex problem because $\phi(\bullet)$ and $R(\bullet)$ are convex with regard to z_L. Therefore, Equation (2.5) can be solved by Fast Iterative Soft-Thresholding Algorithm(FISTA) [2].

3. Update $\overline{W}_l^{k+1}$

The variables $\overline{W}_l^{k+1}(l = 1, \cdots, L)$ are updated as follows:

$$\overline{W}_l^{k+1} \leftarrow \arg\min_{W_l} L_\rho(\{W_i^k\}_{i=1}^{l-1}, W_l, \{\overline{W}_i^{k+1}\}_{i=l+1}^{L}, \overline{\mathbf{z}}_l^{k+1}, \overline{\mathbf{a}}_l^{k+1}, u^k)$$

which is equivalent to the following form:

$$\overline{W}_l^{k+1} \leftarrow \arg\min_{W_l} \phi(\{W_i^k\}_{i=1}^{l-1}, W_l, \{\overline{W}_i^{k+1}\}_{i=l+1}^{L}, \overline{\mathbf{z}}_l^{k+1}, \overline{\mathbf{a}}_l^{k+1}, u^k) \tag{2.6}$$

Due to the same challenge in updating $\overline{a}_l^{k+1}$, we define $\overline{P}_l(W_l; \overline{\theta}_l^{k+1})$ as a quadratic approximation of ϕ at W_l^k. The quadratic approximation is mathematically reformu-

lated as follows [2]:

$$\overline{P}_l(W_l;\overline{\theta}_l^{k+1}) = \phi(\overline{\mathbf{W}}_{l+1}^{k+1},\overline{\mathbf{z}}_l^{k+1},\overline{\mathbf{a}}_l^{k+1},u^k) + (\nabla_{W_l^k}\phi)^T(\overline{\mathbf{W}}_{l+1}^{k+1},\overline{\mathbf{z}}_l^{k+1},\overline{\mathbf{a}}_l^{k+1},u^k)(W_l - W_l^k) + \|\overline{\theta}_l^{k+1}\circ(W_l - W_l^k)^{\circ 2}\|_1/2$$

where $\overline{\theta}_l^{k+1} > 0$ is a parameter vector, which is chosen by the Algorithm 3. Instead of minimizing the Equation (2.6), we minimize the following:

$$\overline{W}_l^{k+1} \leftarrow \arg\min_{W_l} \overline{P}_l(W_l;\overline{\theta}_l^{k+1}) \tag{2.7}$$

Equation (2.7) is convex and hence can be solved exactly.

Algorithm 3: The Backtracking Algorithm to update $\overline{W}_l^{k+1}$

Require: $\overline{\mathbf{W}}_{l+1}^{k+1}$, $\overline{\mathbf{z}}_l^{k+1}$,$\overline{\mathbf{a}}_l^{k+1}$,$u^k$, ρ, some constant $\overline{\gamma} > 1$.
Ensure: $\overline{\theta}_l^{k+1}$,$\overline{W}_l^{k+1}$.
1: Pick up $\overline{\alpha}$ and $\overline{\zeta} = W_l^k - \nabla_{W_l^k}\phi/\overline{\alpha}$.
2: **while** $\phi(\{W_i^k\}_{i=1}^{l-1},\overline{\zeta},\{\overline{W}_i^{k+1}\}_{i=l+1}^{L},\overline{\mathbf{z}}_l^{k+1},\overline{\mathbf{a}}_l^{k+1},u^k) > \overline{P}_l(\overline{\zeta};\overline{\alpha})$ **do**
3: $\overline{\alpha} \leftarrow \overline{\alpha}\,\overline{\gamma}$.
4: Solve $\overline{\zeta}$ by Equation (2.7).
5: **end while**
6: Output $\overline{\theta}_l^{k+1} \leftarrow \overline{\alpha}$.
7: Output $\overline{W}_l^{k+1} \leftarrow \overline{\zeta}$.

4. Update W_l^{k+1}

The variables $W_l^{k+1}(l = 1,\cdots,L)$ are updated as follows:

$$W_l^{k+1} \leftarrow \arg\min_{W_l} L_\rho(\{W_i^{k+1}\}_{i=1}^{l-1},W_l,\{\overline{W}_i^{k+1}\}_{i=l+1}^{L},\mathbf{z}_{l-1}^{k+1},\mathbf{a}_{l-1}^{k+1},u^k)$$

which is equivalent to

$$W_l^{k+1} \leftarrow \arg\min_{W_l} \phi(\{W_i^{k+1}\}_{i=1}^{l-1},W_l,\{\overline{W}_i^{k+1}\}_{i=l+1}^{L},\mathbf{z}_{l-1}^{k+1},\mathbf{a}_{l-1}^{k+1},u^k) \tag{2.8}$$

Similarly, we define $P_l(W_l;\theta_l^{k+1})$ as a quadratic approximation of ϕ at $\overline{W}_l^{k+1}$. The quadratic approximation is then mathematically reformulated as follows [2]:

$$\begin{aligned} P_l(W_l;\theta_l^{k+1}) = \phi(&\mathbf{W}_{l-1}^{k+1},\mathbf{z}_{l-1}^{k+1},\mathbf{a}_{l-1}^{k+1},u^k) \\ &+ (\nabla_{\overline{W}_l^{k+1}}\phi)^T(\mathbf{W}_{l-1}^{k+1},\mathbf{z}_{l-1}^{k+1},\mathbf{a}_{l-1}^{k+1},u^k)(W_l-\overline{W}_l^{k+1}) \\ &+ \|\theta_l^{k+1}\circ(W_l-\overline{W}_l^{k+1})^{\circ 2}\|_1/2 \end{aligned}$$

where $\theta_l^{k+1}>0$ is a parameter vector. Instead of minimizing the Equation (2.8), we minimize the following:

$$W_l^{k+1}\leftarrow \arg\min_{W_l} P_l(W_l;\theta_l^{k+1}) \tag{2.9}$$

The choice of θ_l^{k+1} is discussed in Section A.1 in the Appendix.

5. Update z_l^{k+1}

The variables $z_l^{k+1}(l=1,\cdots,L)$ are updated as follows:

$$z_l^{k+1}\leftarrow \arg\min_{z_l} L_\rho(\mathbf{W}_l^{k+1}\{z_i^{k+1}\}_{i=1}^{l-1},z_l,\{\overline{z}_i^{k+1}\}_{i=l+1}^{L},\mathbf{a}_{l-1}^{k+1},u^k)$$

which is equivalent to the following forms for $z_l(l=1,\cdots,L-1)$:

$$z_l^{k+1}\leftarrow \arg\min_{z_l} \phi(\mathbf{W}_l^{k+1},\{z_i^{k+1}\}_{i=1}^{l-1},z_l,\{\overline{z}_i^{k+1}\}_{i=l+1}^{L-1},\mathbf{a}_{l-1}^{k+1},u^k) \tag{2.10}$$

and for z_L:

$$z_L^{k+1}\leftarrow \arg\min_{z_L} \phi(\mathbf{W}_L^{k+1},\{z_i^{k+1}\}_{i=1}^{L-1},z_L,\mathbf{a}_{L-1}^{k},u^k)+R(z_L,y) \tag{2.11}$$

Solving Equations (2.10) and (2.11) proceeds exactly the same as solving Equations (2.4) and (2.5), respectively.

6. Update a_l^{k+1}

The variables $a_l^{k+1}(l = 1, \cdots, L-1)$ are updated as follows:

$$a_l^{k+1} \leftarrow \arg\min_{a_l} L_\rho(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \{a_i^k\}_{i=1}^{l-1}, a_l, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k)$$

which is equivalent to the following form:

$$a_l^{k+1} \leftarrow \arg\min_{a_l} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \{a_i^k\}_{i=1}^{l-1}, a_l, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k)$$

$Q_l(a_l; \tau_l^{k+1})$ is defined as the quadratic approximation of ϕ at a_l^{k+1} as follows:

$$\begin{aligned} Q_l(a_l; \tau_l^{k+1}) = \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) + (\nabla_{\overline{a}_l^{k+1}}\phi)^T(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k)(a_l - \overline{a}_l^{k+1}) \\ + \|\tau_l^{k+1} \circ (a_l - \overline{a}_l^{k+1})^{\circ 2}\|_1/2 \end{aligned}$$

and we can solve the following problem instead:

$$a_l^{k+1} \leftarrow \arg\min_{a_l} Q_l(a_l; \tau_l^{k+1}) \tag{2.12}$$

where $\tau_l^{k+1} > 0$ is a parameter vector. The solution to Equation (2.12) can be obtained by $a_l^{k+1} \leftarrow \overline{a}_l^{k+1} - \nabla_{\overline{a}_l^{k+1}}\phi/\tau_l^{k+1}$. The choice of an appropriate τ_l^{k+1} is shown in Section A.1 in the Appendix.

2.4 Convergence Analysis

In this section, the theoretical convergence of the proposed dlADMM algorithm is analyzed. Before we formally present the convergence results of the Proposed dlADMM algorithms, Section 2.4.1 presents the necessary assumptions to guarantee the global convergence of dlADMM. In Section 2.4.2, we prove the global convergence of the Proposed dlADMM algorithm.

2.4.1 Assumptions

Assumption 1 (Closed-form Solution)**.** There exist activation functions $a_l = f_l(z_l)$ such that Equations (2.4) and (2.10) have closed form solutions $\overline{z}_l^{k+1} = \overline{h}(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{a}}_l^{k+1})$ and $z_l^{k+1} = h(\mathbf{W}_l^{k+1}, \mathbf{a}_{l-1}^{k+1})$, respectively, where $\overline{h}(\bullet)$ and $h(\bullet)$ are continuous functions.

This assumption can be satisfied by commonly used activation functions such as ReLU and leaky ReLU. For example, for the ReLU function $a_l = \max(z_l, 0)$, Equation (2.10) has the following solution:

$$z_l^{k+1} = \begin{cases} \min(W_l^{k+1} a_{l-1}^{k+1}, 0) & z_l^{k+1} \leq 0 \\ \max((W_l^{k+1} a_{l-1}^{k+1} + \overline{a}_l^{k+1})/2, 0) & z_l^{k+1} \geq 0 \end{cases}$$

Assumption 2 (Objective Function)**.** $F(\mathbf{W}, \mathbf{z}, \mathbf{a})$ is coercive over the nonempty set $G = \{(\mathbf{W}, \mathbf{z}, \mathbf{a}) : z_L - W_L a_{L-1} = 0\}$. In other words, $F(\mathbf{W}, \mathbf{z}, \mathbf{a}) \to \infty$ if $(\mathbf{W}, \mathbf{z}, \mathbf{a}) \in G$ and $\|(\mathbf{W}, \mathbf{z}, \ \mathbf{a})\| \to \infty$. Moreover, $R(z_L, y)$ is Lipschitz differentiable with Lipschitz constant $H \geq 0$.

The Assumption 2 is mild enough for most common loss functions to satisfy. For example, the cross-entropy and square loss are Lipschitz differentiable.

2.4.2 Key Properties

We present the main convergence result of the proposed dlADMM algorithm in this section. Specifically, as long as Assumptions 1-2 hold, then Properties 1-3 are satisfied, which are important to prove the global convergence of the proposed dlADMM algorithm. The proof details are included in Section A.3 and Section A.4 in the Appendix.

Property 1 (Boundness)**.** If $\rho > \frac{\sqrt{17}+1}{2}H$, then $\{\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k\}$ is bounded, and $L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$ is lower bounded.

Property 1 concludes that all variables and the value of L_ρ have lower bounds. It is proven under Assumptions 1 and 2, and its proof can be found in Section A.3 in the Appendix.

Property 2 (Sufficient Descent)**.** If $\rho > \frac{\sqrt{17}+1}{2}H$ so that $C_1 = \rho/2 - H/2 - 2H^2/\rho > 0$ and $C_2 = \rho/2 - 2H^2/\rho > 0$, then

$$\begin{aligned}
&L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k) - L_\rho(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&\geq \sum\nolimits_{l=1}^{L} (\|\overline{\theta}_l^{k+1} \circ (\overline{W}_l^{k+1} - W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{k+1} \circ (\overline{W}_l^{k+1} - \overline{W}_l^{k+1})^{\circ 2}\|_1/2 \\
&+ \|\overline{\tau}_l^{k+1} \circ (\overline{a}_l^{k+1} - a_l^k)^{\circ 2}\|_1/2 + \|\tau_l^{k+1} \circ (a_l^{k+1} - \overline{a}_l^{k+1})^{\circ 2}\|_1/2) \\
&+ C_2\|\overline{z}_L^{k+1} - z_L^k\|_2^2 + C_1\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2
\end{aligned} \tag{2.13}$$

Property 2 depicts the monotonic decrease of the objective value during iterations. The proof of Property 2 is detailed in Section A.3 in the Appendix.

Property 3 (Subgradient Bound)**.** There exist a bounded $C^{k+1} > 0$ and $g \in \partial L_\rho(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1})$ such that

$$\|g\| \leq C^{k+1}(\|\mathbf{W}^{k+1} - \overline{\mathbf{W}}^{k+1}\| + \|\mathbf{z}^{k+1} - \overline{\mathbf{z}}^{k+1}\| + \|\mathbf{a}^{k+1} - \overline{\mathbf{a}}^{k+1}\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|) \tag{2.14}$$

Property 3 ensures that the subgradient of the objective function is bounded by variables. The proof of Property 3 requires Property 1 and the proof is elaborated in Section A.3 in the Appendix. Now the global convergence of the Proposed dlADMM algorithm is presented. The following theorem states that Properties 1-3 are guaranteed.

Theorem 1. For any $\rho > \frac{\sqrt{17}+1}{2}H$, if Assumptions 1 and 2 are satisfied, then Properties 1-3 hold.

Proof. This theorem can be concluded by the proofs in Section A.3 in the Appendix. □

The next theorem presents the global convergence of the Proposed dlADMM algorithm.

Theorem 2 (Global Convergence)**.** If $\rho > \frac{\sqrt{17}+1}{2}H$ and $\{\overline{\theta}_l^{k+1}\}_{l=1}^{L}, \{\theta_l^{k+1}\}_{l=1}^{L}, \{\overline{\tau}_l^{k+1}\}_{l=1}^{L-1}$, and $\{\tau_l^{k+1}\}_{l=1}^{L-1}$ are bounded, then for the variables $(\mathbf{W}, \mathbf{z}, \mathbf{a}, u)$ in Problem 2, starting from any $(\mathbf{W}^0, \mathbf{z}^0, \mathbf{a}^0, u^0)$, it has at least a limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$, and any limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$ is a stationary point of Problem 2(i.e. $0 \in \partial L_\rho(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$). Or equivalently,

$$z_L^* = W_L^* a_{L-1}^* \qquad 0 \in \partial_{\mathbf{W}^*} L_\rho(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$$
$$0 \in \partial_{\mathbf{z}^*} L_\rho(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*) \qquad \nabla_{\mathbf{a}^*} L_\rho(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*) = 0$$

Proof. Because $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$ is bounded, there exists a subsequence $(\mathbf{W}^s, \mathbf{z}^s, \mathbf{a}^s, \mathbf{u}^s)$ such that $(\mathbf{W}^s, \mathbf{z}^s, \mathbf{a}^s, u^s) \to (\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$ where $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$ is a limit point. By Properties 1 and 2, $L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$ is non-increasing and lower bounded and hence converges. By Property 2, we prove that $\|\overline{\mathbf{W}}^{k+1} - \mathbf{W}^k\| \to 0$, $\|\overline{\mathbf{a}}^{k+1} - \mathbf{a}^k\| \to 0$, $\|\mathbf{W}^{k+1} - \overline{\mathbf{W}}^{k+1}\| \to 0$, and $\|\mathbf{a}^{k+1} - \overline{\mathbf{a}}^{k+1}\| \to 0$, as $k \to \infty$. Therefore $\|\mathbf{W}^{k+1} - \mathbf{W}^k\| \to 0$, and $\|\mathbf{a}^{k+1} - \mathbf{a}^k\| \to 0$, as $k \to \infty$. Moreover, from Assumption 1, we know that $\overline{\mathbf{z}}^{k+1} \to \mathbf{z}^k$ and $\mathbf{z}^{k+1} \to \overline{\mathbf{z}}^{k+1}$ as $k \to \infty$. Therefore, $\mathbf{z}^{k+1} \to \mathbf{z}^k$. We infer there exists $g^k \in \partial L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$ such that $\|g^k\| \to 0$ as $k \to \infty$ based on Property 3. Specifically, $\|g^s\| \to 0$ as $s \to \infty$. According to the definition of general subgradient(Defintion 8.3 in [84]), we have $0 \in \partial L_\rho(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$. In other words, the limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*, u^*)$ is a stationary point of L_ρ defined in Equation (2.1). □

Theorem 2 shows that our dlADMM algorithm converges globally for sufficiently large ρ, which is consistent with previous literature [51, 107]. The next theorem shows that the Proposed dlADMM converges globally with a sublinear convergence rate $o(1/k)$, where k is the number of iterations.

Theorem 3 (Convergence Rate)**.** For a sequence $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$, define $c_k = \min_{0\leq i\leq k}(\sum_{l=1}^{L}(\|\overline{\theta}_l^{i+1} \circ (\overline{W}_l^{i+1} - W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{i+1} \circ (\overline{W}_l^{i+1} - \overline{W}_l^{i+1})^{\circ 2}\|_1/2) + \sum_{l=1}^{L-1}(\|\overline{\tau}_l^{i+1} \circ (\overline{a}_l^{i+1} - a_l^i)^{\circ 2}\|_1/2 + \|\tau_l^{i+1} \circ (a_l^{i+1} - \overline{a}_l^{i+1})^{\circ 2}\|_1/2) + C_2\|\overline{z}_L^{i+1} - z_L^i\|_2^2 + C_1\|z_L^{i+1} - \overline{z}_L^{i+1}\|_2^2)$, then the convergence rate of c_k is $o(1/k)$.

Proof. The proof of this theorem is included in Section A.4 in the Appendix. □

2.5 Experiments

In this section, we evaluate the dlADMM algorithm using benchmark datasets. The effectiveness, efficiency, and convergence properties of dlADMM are compared with state-of-the-art methods. All experiments were conducted on 64-bit Ubuntu16.04 LTS with Intel(R) Xeon processor and GTX1080Ti GPU.

2.5.1 Experiment Setup

Dataset

In this experiment, two benchmark datasets were used for performance evaluation: MNIST [58] and Fashion MNIST [112]. The MNIST dataset has ten classes of handwritten-digit images, which was firstly introduced by Lecun et al. in 1998 [58]. It contains 55,000 training samples and 10,000 test samples with 784 features each, which is provided by the Keras library [18]. Unlike the MNIST dataset, the Fashion MNIST dataset has ten classes of assortment images on the website of Zalando, which is Europe's largest online fashion platform [112]. The Fashion-MNIST dataset consists of 60,000 training samples and 10,000 test samples with 784 features each.

Experiment Settings

We set up a network architecture that contained two hidden layers with $1,000$ hidden units each. The ReLU was used for the activation function for both network structures. The loss function was set as the deterministic cross-entropy loss. ν was set to 10^{-6}. ρ was initialized as 10^{-6} and was multiplied by 10 every 100 iterations. The number of iterations was set to 200. In the experiment, one iteration means one epoch.

Comparison Methods

Since this chapter focuses on the MLP model, GD and its variants and ADMM are state-of-the-art methods and hence served as comparison methods. For GD-based methods, the full batch dataset is used for training models. All parameters were chosen by the accuracy of the training dataset. The baselines are described as follows:

1. Gradient Descent(GD) [4]. The GD and its variants are the most popular deep learning optimizers, whose convergence has been studied extensively in the literature. The learning rate of GD was set to 10^{-6} for both the MNIST and Fashion MNIST datasets.

2. Adaptive gradient algorithm(Adagrad) [28]. Adagrad is an improved version of GD: rather than fixing the learning rate during iteration, it adapts the learning rate to the hyperparameter. The learning rate of Adagrad was set to 10^{-3} for both the MNIST and Fashion MNIST datasets.

3. Adaptive learning rate method(Adadelta) [118]. As an improved version of the Adagrad, the Adadelta is proposed to overcome the sensitivity to hyperparameter selection. The learning rate of Adadelta was set to 0.1 for both the MNIST and Fashion MNIST datasets.

4. Adaptive momentum estimation(Adam) [52]. Adam is the most popular optimization method for deep learning models. It estimates the first and second momen-

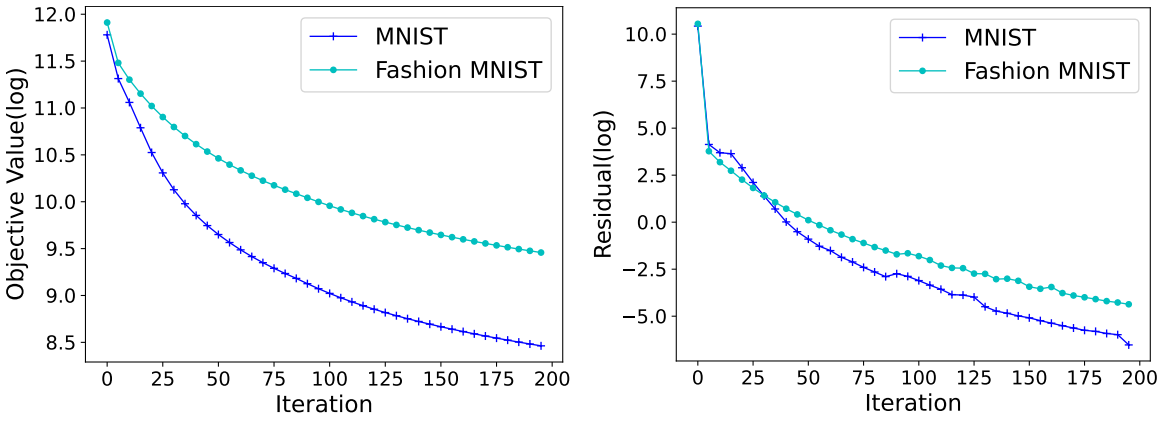
tum to correct the biased gradient and thus makes convergence fast. The learning rate of Adam was set to 10^{-3} for both the MNIST and Fashion MNIST datasets.

5. Alternating Direction Method of Multipliers(ADMM) [93]. ADMM is a powerful convex optimization method because it can split an objective function into a series of subproblems, which are coordinated to get global solutions. It is scalable to large-scale datasets and supports parallel computations. The ρ of ADMM was set to 1 for both the MNIST and Fashion MNIST datasets.

2.5.2 Experimental Results

In this section, experimental results of the Proposed dlADMM algorithm are analyzed against comparison methods.

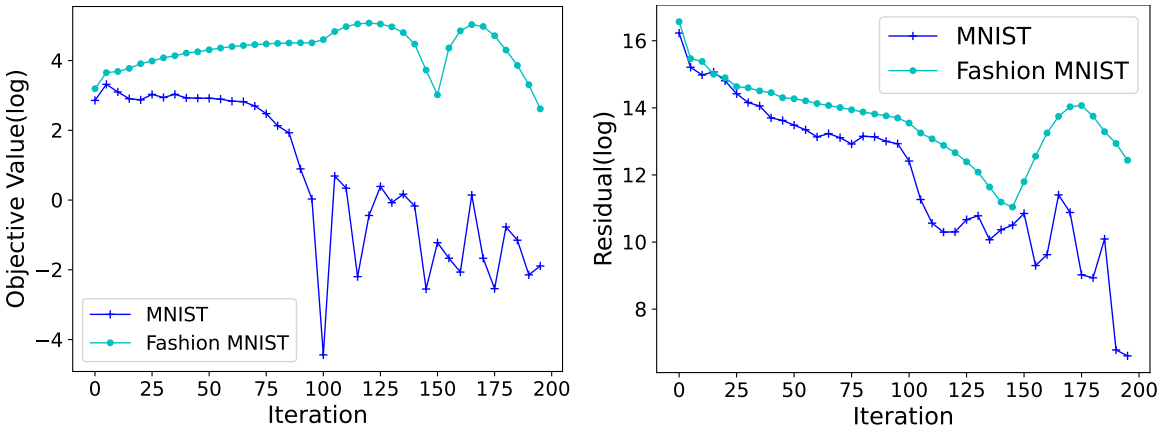
Convergence



(a). Objective value (b). Residual

Figure 2.2: Convergence curves of the proposed dlADMM algorithm on two datasets when $\rho = 1$.

First, we show that our proposed dlADMM algorithm converges when ρ is sufficiently large and diverges when ρ is small for both the MNIST dataset and the Fashion MNIST dataset.



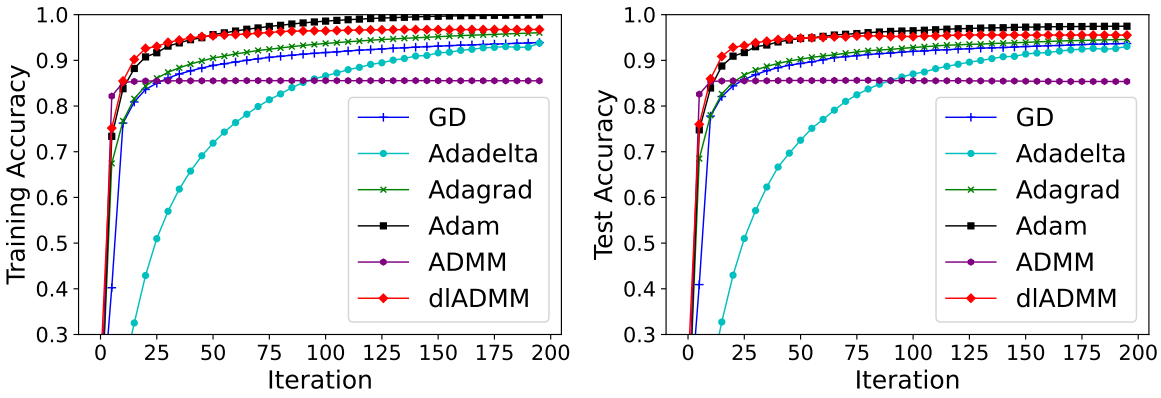
(a). Objective value (b). Residual

Figure 2.3: Divergence curves of the proposed dlADMM algorithm on two datasets when $\rho = 10^{-6}$.

The convergence and divergence of dlADMM algorithm are shown in Figures 2.2 and 2.3 when $\rho = 1$ and $\rho = 10^{-6}$,respectively. In Figures 2.2(a) and 2.3(a), the X axis and Y axis denote the number of iterations and the logarithm of objective value, respectively. In Figures, 2.2(b) and 2.3(b), the X-axis and Y-axis denote the number of iterations and the logarithm of the residual, respectively. Figure 2.2, both the objective value and the residual decreased monotonically for the MNIST dataset and the Fashion-MNIST dataset, which validates our theoretical guarantees in Theorem 2. Moreover, Figure 2.3 illustrates that both the objective value and the residual diverge when $\rho = 10^{-6}$. The curves fluctuated drastically on the objective value. Even though there was a decreasing trend for the residual, it still fluctuated irregularly and failed to converge.

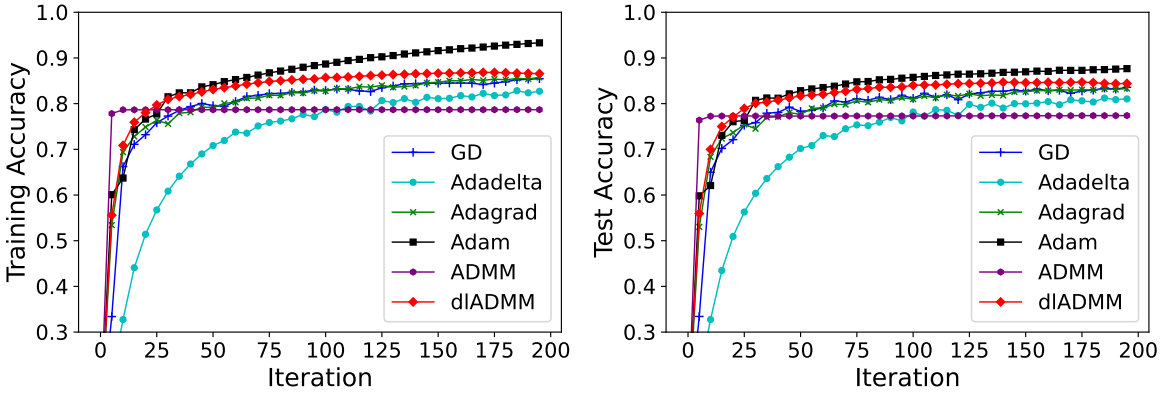
Performance

Figure 2.4 and Figure 2.5 show the curves of the training accuracy and test accuracy of our proposed dlADMM algorithm and baselines, respectively. Overall, both the training accuracy and the test accuracy of our proposed dlADMM algorithm outperformed most baselines for both the MNIST dataset and the Fashion MNIST dataset.



(a). Training accuracy (b). Test accuracy

Figure 2.4: Performance of the proposed dlADMM algorithm against all comparison methods on the MNIST dataset.



(a). Training Accuracy (b).Test Accuracy

Figure 2.5: Performance of the proposed dlADMM algorithm against all comparison methods on the Fashion MNIST dataset.

Specifically, the curves of our dlADMM algorithm soared to 0.8 at the early stage and then raised steadily towards 0.9 or more. The curves of the most GD-related methods, GD, Adadelta, and Adagrad, moved more slowly than our proposed dlADMM algorithm. The curves of the ADMM also rocketed to around 0.8 but decreased slightly later on. Only the state-of-the-art Adam performed better than dlADMM slightly.

Scalability Analysis

In this subsection, the relationship between running time per iteration of our proposed dlADMM algorithm and three potential factors, namely, the value of ρ, the size of

MNIST dataset: From 200 to 1,000 neurons					
ρ \ neurons	200	400	600	800	1000
10^{-6}	1.9025	2.7750	3.6615	4.5709	5.7988
10^{-5}	2.8778	4.6197	6.3620	8.2563	10.0323
10^{-4}	2.2761	3.9745	5.8645	7.6656	9.9221
10^{-3}	2.4361	4.3284	6.5651	8.7357	11.3736
10^{-2}	2.7912	5.1383	7.8249	10.0300	13.4485
Fashion MNIST dataset: From 200 to 1,000 neurons					
ρ \ neurons	200	400	600	800	1000
10^{-6}	2.0069	2.8694	4.0506	5.1438	6.7406
10^{-5}	3.3445	5.4190	7.3785	9.0813	11.0531
10^{-4}	2.4974	4.3729	6.4257	8.3520	10.0728
10^{-3}	2.7108	4.7236	7.1507	9.4534	12.3326
10^{-2}	2.9577	5.4173	8.2518	10.0945	14.3465

Table 2.1: The relationship between running time per iteration(in second) of the proposed dlADMM algorithm and the number of neurons as well as the value of ρ.

MNIST dataset: From 11,000 to 55,000 training samples					
ρ \ size	11,000	22,000	33,000	44,000	55,000
10^{-6}	1.0670	2.0682	3.3089	4.6546	5.7709
10^{-5}	2.3981	3.9086	6.2175	7.9188	10.2741
10^{-4}	2.1290	3.7891	5.6843	7.7625	9.8843
10^{-3}	2.1295	4.1939	6.5039	8.8835	11.3368
10^{-2}	2.5154	4.9638	7.6606	10.4580	13.4021
Fashion MNIST dataset: From 12,000 to 60,000 training samples					
ρ \ size	12,000	24,000	36,000	48,000	60,000
10^{-6}	1.2163	2.3376	3.7053	5.1491	6.7298
10^{-5}	2.5772	4.3417	6.6681	8.3763	11.0292
10^{-4}	2.3216	4.1163	6.2355	8.3819	10.7120
10^{-3}	2.3149	4.5250	6.9834	9.5853	12.3232
10^{-2}	2.7381	5.3373	8.1585	11.1992	14.2487

Table 2.2: The relationship between running time per iteration(in second) of the proposed dlADMM algorithm and the size of training samples as well as the value of ρ.

training samples, and the number of neurons was explored. The running time was calculated by the average of 200 iterations.

Firstly, when the training size was fixed, the computational result for the MNIST dataset and Fashion MNIST dataset is shown in Table 2.1. The number of neurons for each layer ranged from 200 to 1,000, with an increase of 200 each time. The value of ρ ranged from 10^{-6} to 10^{-2}, multiplying by 10 each time. Generally, the running time increased as the number of neurons and the value of ρ became larger. However, there were a few exceptions: for example, when there were 200 neurons for the MNIST dataset, and ρ increased from 10^{-5} to 10^{-4}, the running time per iteration dropped from 2.8778 seconds to 2.2761 seconds.

Secondly, we fixed the number of neurons for each layer as $1,000$. The relationship between running time per iteration, the training size, and the value of ρ is shown in Table 2.2. The value of ρ ranged from 10^{-6} to 10^{-2}, multiplying by 10 each time. The training size of the MNIST dataset ranged from $11,000$ to $55,000$, with an increase of $11,000$ each time. The training size of the Fashion MNIST dataset ranged from $12,000$ to $60,000$, with an increase of $12,000$ each time. Similar to Table 2.2, the running time increased generally as the training sample and the value of ρ became larger and some exceptions exist.

2.6 Conclusion

The ADMM is a good alternative to GD for deep learning problems. In this chapter, we propose a novel dlADMM algorithm to address some previously mentioned challenges. Firstly, the Proposed dlADMM updates parameters from backward to forward to transmit parameter information more efficiently. The time complexity is successfully reduced from $O(n^3)$ to $O(n^2)$ by iterative quadratic approximations and backtracking. Finally, the Proposed dlADMM is guaranteed to converge to a station-

ary solution under mild conditions. Experiments on benchmark datasets demonstrate that our proposed dlADMM algorithm outperformed most of the comparison methods.

Chapter 3

The mDLAM Algorithm

3.1 Introduction

In Chapter 2, we proposed a novel ADMM-based algorithm to train the MLP training problem efficiently with convergence guarantees. We extend this idea to Alternating Minimization(AM) methods in this chapter. However, as an emerging domain, AM for deep model optimization suffers from several unsolved challenges including **1. Convergence properties are sensitive to penalty parameters.** Even though the dlADMM algorithm is guaranteed convergence in the MLP problem, such convergence guarantee is dependent on the choice of penalty hyperparameters: the convergence cannot be guaranteed anymore when penalty hyperparameters are small. **2. Slow convergence rate.** To the best of our knowledge, almost all existing AM methods can only achieve a sublinear convergence rate. For example, the convergence rate of the ADMM and the BCD is proven to be $O(1/k)$, where k is the number of iterations [101, 100, 120]. Therefore, there is still a lack of a theoretical framework that can achieve a faster convergence rate.

To simultaneously address these technical problems, we propose a new formulation of the neural network problem, along with a novel monotonous Deep Learning Alter-

nating Minimization(mDLAM) algorithm. Specifically, we, for the first time, transform the original neural network optimization problem into an inequality-constrained problem that can infinitely approximate the original one. Applying this innovation to an inequality-constraint-based transformation not only ensures the convexity and hence easily ensures global minima of all subproblems but also prevents the output of a nonlinear function from changing much and reduces sensitivity to the input. Moreover, our proposed mDLAM algorithm can achieve a linear convergence rate theoretically, and the choice of hyperparameters does not affect the convergence of our mDLAM algorithm theoretically. Extensive experiments on four benchmark datasets show the convergence, effectiveness, and efficiency of the proposed mDLAM algorithm. Our contributions in this chapter include:

- We propose a novel formulation for neural network optimization. The deeply nested activation functions are disentangled into separate functions innovatively coordinated by inherently convex inequality constraints.

- We present an efficient optimization algorithm. A quadratic approximation technique is utilized to avoid matrix inversion. Every subproblem has a closed-form solution. The Nesterov acceleration technique is applied to further boost convergence.

- We investigate the convergence of the proposed mDLAM algorithm under mild conditions. The new mDLAM algorithm is guaranteed to converge to a stationary point whatever hyperparameters we choose. Furthermore, the proposed mDLAM algorithm is shown to achieve a linear convergence rate, which is faster than existing methods.

- Extensive experiments have been conducted to demonstrate the effectiveness of the proposed mDLAM algorithm. We test our proposed mDLAM algorithm on four benchmark datasets. Experimental results illustrate that our proposed

mDLAM algorithm is linearly convergent on four datasets, and outperforms consistently state-of-the-art optimizers. Sensitivity analysis of the running time shows that it increases linearly with the increase of neurons and hyperparameters.

The rest of this chapter is organized as follows: In Section 3.2, we summarize recent related research work to this chapter. In Section 3.3, we formulate the MLP training problem and present the proposed mDLAM algorithm to train the MLP model. Section 3.4 details convergence properties of the proposed mDLAM algorithm. Extensive experiments on benchmark datasets are shown in Section 3.5, and Section 3.6 concludes this work.

3.2 Related Work

All existing works on deep learning optimization methods fall into two major categories: GD methods and AM methods, which are shown as follows:

GD Methods: The renaissance of GD can be traced back to 1951 when Robbins and Monro published the first paper [83]. The famous back-propagation algorithm was introduced by Rumelhart et al. [85]. Many variants of GD methods have since been presented, including the use of Polyak momentum, which accelerates the convergence of iterative methods [80], and research by Sutskever et al., who highlighted the importance of Nesterov momentum and initialization [91]. During the last decade, many well-known GD methods which are incorporated with adaptive learning rates have been proposed by the deep learning community, which include but are not limited to AdaGrad [28], RMSProp [95], Adam [52], AMSGrad [82], Adabelief [128] and Adabound [65].

Applications of AM Methods for Deep Learning: Many recent works have applied AM algorithms to specific deep learning applications. For example, Taylor et

al. presented the ADMM to solve an MLP training problem via transforming it into an equality-constrained problem, where many subproblems split by ADMM can be solved efficiently [93], Zhang et al. handled Very Deep Supervised Hashing(VDSH) problems by utilizing an ADMM algorithm to overcome issues related to vanishing gradients and poor computational efficiency [125]. Zhang and Bastiaan trained a deep neural network by utilizing ADMM with a graph [121] and Askari et al. introduced a new framework for MLP models and optimize the objective using BCD methods [71]. Li et al. proposed an ADMM algorithm to achieve distributed learning of Graph Convolutional Network(GCN) via community detection [61]. Qiao et al. proposed an inertial proximal AM method to train MLP models [81].

Convergence of AM Methods for Deep Learning: Aside from applications, the other branch of works mathematically proves the convergence of the proposed AM approaches. For instance, Carreira and Wang proposed a method involving the use of auxiliary coordinates to replace a nested neural network with a constrained problem without nesting [9]. Lau et al. proposed a BCD optimization framework and proved the convergence via the Kurdyka-Lojasiewicz(KL) property [57], while Choromanska et al. proposed a BCD algorithm for training deep MLP models based on the concept of co-activation memory [19], and a BCD algorithm with R-linear convergence was proposed by Zhang and Brand to train Tikhonov regularized deep neural networks [124]. Jagatap and Hegde introduced a new family of AM methods and prove their convergence to a global minimum [47]. Yu et al. proved the convergence of the proposed ADMM for RNN models [92]. However, to the best of our knowledge, there is a lack of a flexible framework that allows for different activation functions and guarantees a linear convergence rate.

3.3 Model and Algorithm

3.3.1 Inequality Approximation for Deep Learning

In this section, we propose our problem transformation to address Problem 1. The equality constraint $a_l = f_l(z_l)$ is the most challenging one to handle here because common activation functions such as sigmoid [31] are nonlinear. This makes them nonconvex constraints and hence it is difficult to obtain the optimal solution when solving the z_l-subproblem [93]. Moreover, there is no guarantee for AM methods to solve the nonlinear equality constrained Problem 1 [99]. To deal with these two challenges, the following assumption is required for problem transformation:

Assumption 3. $f_l(z_l)$ $(l = 1, \ldots, n)$ are quasilinear.

The quasilinearity is defined in Section B.1 in the Appendix. Assumption 3 is so mild that most of the widely used nonlinear activation functions satisfy it, including tanh [117], sigmoid [31], and the Rectified Linear Unit(ReLU) [66]. Then we innovatively transform the original nonconvex constraints into inequality constraints, which can be an infinite approximation of Problem 1. To do this, we introduce a tolerance $\varepsilon > 0$ and reformulate Problem 1 to the following:

$$\begin{aligned} &\min_{W_l, z_l, a_l} R(z_L, y) \\ &s.t.\ z_l = W_l a_{l-1} (l = 1, \cdots, L),\ f_l(z_l) - \varepsilon \leq a_l \leq f_l(z_l) + \varepsilon (l = 1, \cdots, L-1). \end{aligned}$$

For the linear constraint $z_l = W_l a_{l-1}$, this can be transformed into a penalty term in the objective function to minimize the difference between z_l and $W_l a_{l-1}$. The formulation is shown as follows:

Problem 3.

$$\min_{W_l,z_l,a_l} F(\mathbf{W},\mathbf{z},\mathbf{a}) = R(z_L,y) + \sum_{l=1}^{L} \phi(a_{l-1},W_l,z_l),$$
$$s.t.\ f_l(z_l) - \varepsilon \le a_l \le f_l(z_l) + \varepsilon (l = 1,\cdots,L-1).$$

The penalty term is defined as $\phi(a_{l-1},W_l,z_l) = \frac{\rho}{2}\|z_l - W_l a_{l-1}\|_2^2$, where $\rho > 0$ a penalty parameter. $\mathbf{W} = \{W_l\}_{l=1}^{L}$, $\mathbf{z} = \{z_l\}_{l=1}^{L}$,$\mathbf{a} = \{a_l\}_{l=1}^{L-1}$. As $\rho \to \infty$ and $\varepsilon \to 0$, Problem 3 approaches Problem 1.

The introduction of ε is to project the nonconvex constraints to ε-balls, thus transforming the nonconvex Problem 1 into Problem 3. Even though Problem 3 is still nonconvex because $f_l(z_l)$ can be nonconvex(e.g. tanh and smooth sigmoid), it is convex concerning one variable when others are fixed(i.e. multi-convex), which is much easier to solve by AM [115]. For example, Problem 3 is convex with regard to $\mathbf{z}$ when $\mathbf{W}$, and $\mathbf{a}$ are fixed.

3.3.2 Alternating Optimization

We present the mDLAM algorithm to solve Problem 3 in this section. A potential challenge to solving Problem 3 is a slow theoretical convergence rate. For example, the convergence rate of the proposed dlADMM algorithm in Chapter 2 to solve Problem 3 is sublinear $o(1/k)$, where k is the number of iterations [101]. To address this challenge, we apply the famous Nesterov acceleration technique to boost the convergence of our proposed mDLAM algorithm, and we prove its linear convergence theoretically in the next section.

Algorithm 4 shows our proposed mDLAM algorithm. To simplify the notation, $\mathbf{W}_{\le l}^{k+1} = \{\{W_i^{k+1}\}_{i=1}^{l}, \{W_i^{k}\}_{i=l+1}^{L}\}$, $\mathbf{z}_{\le l}^{k+1} = \{\{z_i^{k+1}\}_{i=1}^{l}, \{z_i^{k}\}_{i=l+1}^{L}\}$ and $\mathbf{a}_{\le l}^{k+1} = \{\{a_i^{k+1}\}_{i=1}^{l}, \{a_i^{k}\}_{i=l+1}^{L-1}\}$. In Algorithm 4, Lines 6, 10, and 21 apply the Nesterov acceleration technique and update W_l, z_l and a_l, respectively. the proposed mDLAM

Algorithm 4: The proposed mDLAM algorithm

Require: y, $a_0 = x$.
Ensure: $a_l, W_l, z_l (l = 1, \cdots, L)$.
Initialize ρ, $k = 0$, $s^0 = 0$.
repeat
  $s^{k+1} \leftarrow \frac{1+\sqrt{1+4(s^k)^2}}{2}$
  $\omega^k \leftarrow \frac{s^k - 1}{s^{k+1}}$
  for $l = 1$ to L **do**
    $\overline{W}_l^{k+1} \leftarrow W_l^k + (W_l^k - W_l^{k-1})\omega^k$ and update W_l^{k+1} in Equation (3.3).
    if $F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq F(\mathbf{W}_{\leq l-1}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$ **then**
      $\overline{W}_l^{k+1} \leftarrow W_l^k$ and update W_l^{k+1} in Equation (3.3).
    end if
    $\overline{z}_l^{k+1} \leftarrow z_l^k + (z_l^k - z_l^{k-1})\omega^k$
    if $l = L$ **then**
      Update z_L^{k+1} in Equation (3.5).
      if $F(\mathbf{W}_{\leq L}^{k+1}, \mathbf{z}_{\leq L}^{k+1}, \mathbf{a}_{\leq L-1}^{k+1}) \geq F(\mathbf{W}_{\leq L}^{k+1}, \mathbf{z}_{\leq L-1}^{k+1}, \mathbf{a}_{\leq L-1}^{k+1})$ **then**
        $\overline{z}_L^{k+1} \leftarrow z_L^k$ and update z_L^{k+1} in Equation (3.5).
      end if
    else
      Update z_l^{k+1} in Equation (3.4).
      if $F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$ **then**
        $\overline{z}_l^{k+1} \leftarrow z_l^k$ and update z_l^{k+1} in Equation (3.4).
      end if
      $\overline{a}_l^{k+1} \leftarrow a_l^k + (a_l^k - a_l^{k-1})\omega^k$ and update a_l^{k+1} in Equation (3.6).
      if $F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l}^{k+1}) \geq F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$ **then**
        $\overline{a}_l^{k+1} \leftarrow a_l^k$ and update a_l^{k+1} in Equation (3.6).
      end if
    end if
  end for
  $k \leftarrow k + 1$.
until convergence.
Output a_l, W_l, z_l.

algorithm guarantees the decrease of objective F: for example, if the updated W_l^{k+1} in Line 7 of Algorithm 4 increases the value of F, i.e. $F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq F(\mathbf{W}_{\leq l-1}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$, then W_l^{k+1} is updated again by setting $\overline{W}_l^{k+1} = W_l^k$ in Line 8 of Algorithm 4, which ensures the decline of F. The same procedure is applied in Lines 13-15, Lines 18-20, and Lines 22-24 in Algorithm 4, respectively.

Next, all subproblems are shown as follows:

1. Update W_l

The variables $W_l (l = 1, \cdots, L)$ are updated as follows:

$$W_l^{k+1} \leftarrow \arg\min_{W_l} \phi(a_{l-1}^{k+1}, W_l, z_l^k). \tag{3.1}$$

Because W_l and a_{l-1} are coupled in $\phi(\bullet)$, solving W_l requires an inversion operation of a_{l-1}^{k+1}, which is computationally expensive. Motivated by the proposed dlADMM algorithm in Chapter 2, we define $P_l^{k+1}(W_l; \theta_l^{k+1})$ as a quadratic approximation of ϕ at W_l^k as follows:

$$P_l^{k+1}(W_l; \theta_l^{k+1}) = \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + (\nabla_{\overline{W}_l^{k+1}} \phi)^T (W_l - \overline{W}_l^{k+1}) + \frac{\theta_l^{k+1}}{2} \|W_l - \overline{W}_l^{k+1}\|_2^2,$$

where $\theta_l^{k+1} > 0$ is a scalar parameter, which can be chosen by the backtracking algorithm [101] to meet the following condition

$$P_l^{k+1}(W_l^{k+1}; \theta_l^{k+1}) \geq \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^k). \tag{3.2}$$

Rather than minimizing Equation (3.1), we instead minimize the following:

$$W_l^{k+1} \leftarrow \arg\min_{W_l} P_l^{k+1}(W_l; \theta_l^{k+1}). \tag{3.3}$$

2. Update z_l

The variables $z_l(l = 1, \cdots, L)$ are updated as follows:

$$z_l^{k+1} \leftarrow \arg\min_{z_l} \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l),\ s.t.\ f_l(z_l) - \varepsilon \leq a_l \leq f_l(z_l) + \varepsilon\ l < L).$$
$$z_L^{k+1} \leftarrow \arg\min_{z_L} \phi(a_{L-1}^{k+1}, W_L^{k+1}, z_L) + R(z_L, y).$$

Similar to updating W_l, we define $V_l^{k+1}(z_l)$ as follows:

$$V_l^{k+1}(z_l) = \phi(a_{l-1}^{k+1}, W_l^{k+1}, \overline{z}_l^{k+1}) + (\nabla_{\overline{z}_l^{k+1}}\phi)^T(z_l - \overline{z}_l^{k+1}) + \frac{\rho}{2}\|z_l - \overline{z}_l^{k+1}\|_2^2.$$

Hence, we solve the following problems:

$$z_l^{k+1} \leftarrow \arg\min_{z_l} V_l^{k+1}(z_l),\ s.t.\ f_l(z_l) - \varepsilon \leq a_l \leq f_l(z_l) + \varepsilon\ (l < L). \tag{3.4}$$
$$z_L^{k+1} \leftarrow \arg\min_{z_L} V_L^{k+1}(z_L) + R(z_L, y). \tag{3.5}$$

As for $z_l(l = 1, \cdots, l-1)$, the solution is

$$z_l^{k+1} \leftarrow \min(\max(B_1^{k+1}, \overline{z}_l^{k+1} - \nabla\phi_{\overline{z}_l^{k+1}}/\rho), B_2^{k+1}),$$

where B_1^{k+1} and B_2^{k+1} represent the lower bound and the upper bound of the set $\{z_l | f_l(z_l) - \varepsilon \leq a_l^k \leq f_l(z_l) + \varepsilon\}$. Equation (3.5) is easy to solve using the Fast Iterative Soft Thresholding Algorithm(FISTA) [2].

3. Update a_l

The variables $a_l(l = 1, \cdots, L-1)$ are updated as follows:

$$a_l^{k+1} \leftarrow \arg\min_{a_l} \phi(a_l, W_{l+1}^k, z_{l+1}^k),\ s.t.\ f_l(z_l^{k+1}) - \varepsilon \leq a_l \leq f_l(z_l^{k+1}) + \varepsilon.$$

Similar to updating W_l^{k+1}, $Q_l^{k+1}(a_l;\tau_l^{k+1})$ is defined as

$$Q_l^{k+1}(a_l;\tau_l^{k+1}) = \phi(\overline{a}_l^{k+1}, W_{l+1}^k, z_{l+1}^k) + (\nabla_{\overline{a}_l^{k+1}}\phi)^T(a_l - \overline{a}_l^{k+1}) + \frac{\tau_l^{k+1}}{2}\|a_l - \overline{a}_l^{k+1}\|_2^2,$$

and this allows us to solve the following problem instead:

$$a_l^{k+1} \leftarrow \arg\min_{a_l} Q_l^{k+1}(a_l;\tau_l^{k+1}), s.t.\ f_l(z_l^{k+1}) - \varepsilon \le a_l \le f_l(z_l^{k+1}) + \varepsilon \tag{3.6}$$

where $\tau_l^{k+1} > 0$ is a scalar parameter, which can be chosen by the backtracking algorithm [101] to meet the following condition:

$$Q_l^{k+1}(a_l^{k+1};\tau_l^{k+1}) \ge \phi(a_l^{k+1}, W_{l+1}^k, z_{l+1}^k).$$

The solution can be obtained by

$$a_l^{k+1} \leftarrow \min(\max(f_l(z_l^{k+1}) - \varepsilon, \overline{a}_l^{k+1} - \nabla_{\overline{a}_l^{k+1}}\phi/\tau_l^{k+1}), f_l(z_l^{k+1}) + \varepsilon).$$

3.4 Convergence Analysis

In this section, the convergence of the proposed algorithm is analyzed. Due to space limit, all proofs are detailed in Section B.2 and Section B.3 in the appendix. The following mild assumption is required for the convergence analysis of the proposed mDLAM algorithm:

Assumption 4. $F(\mathbf{W}, \mathbf{z}, \mathbf{a})$ is coercive over the domain $\{(\mathbf{W}, \mathbf{z}, \mathbf{a}) | f_l(z_l) - \varepsilon \le a_l \le f_l(z_l) + \varepsilon\ (l = 1, \cdots, L-1)\}$.

The coercivity is defined in Section B.1 in the Appendix. Assumption 4 is also mild such that common loss functions such as the least square loss and the cross-entropy loss satisfy it [101].

3.4.1 Convergence Properties

Firstly, the following preliminary lemma is useful to prove the convergence properties of the proposed mDLAM algorithm.

Lemma 1. In Algorithm 4, there exist $\alpha_l^k, \gamma_l^k, \delta_l^k > 0$ such that for $\forall k \in \mathbb{N}$, $W_l^k, z_l^k (l = 1, 2, \cdots, L)$, and $a_l^k (l = 1, 2, \cdots, L-1)$, it holds that

$$F(\mathbf{W}_{\leq l-1}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) - F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq \frac{\alpha_l^{k+1}}{2} \|W_l^{k+1} - W_l^k\|_2^2, \tag{3.7}$$

$$F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) - F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq \frac{\gamma_l^{k+1}}{2} \|z_l^{k+1} - z_l^k\|_2^2, \tag{3.8}$$

$$F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) - F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l}^{k+1}) \geq \frac{\delta^{k+1}}{2} \|a_l^{k+1} - a_l^k\|_2^2. \tag{3.9}$$

It shows that the objective decreases when all variables are updated. Based on Assumption 4 and Lemma 1, three convergence properties hold, which are shown in the following:

Lemma 2 (Objective Decrease)**.** In Algorithm 4, it holds that for any $k \in \mathbb{N}$, $F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) \geq F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1})$. Moreover, F is convergent. That is, $F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) \to F^*$ as $k \to \infty$, where F^* is the convergent value of F.

This lemma guarantees the decrease and hence convergence of the objective.

Lemma 3 (Bounded Objective and Variables)**.** In Algorithm 4, it holds that for any $k \in \mathbb{N}$

(a). $\mathbf{F}(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is upper bounded. Moreover, $\lim_{k\to\infty} \mathbf{W}^{k+1} - \mathbf{W}^k = 0$, $\lim_{k\to\infty} \mathbf{z}^{k+1} - \mathbf{z}^k = 0$, and $\lim_{k\to\infty} \mathbf{a}^{k+1} - \mathbf{a}^k = 0$.

(b). $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is bounded. That is, there exist scalars $M_{\mathbf{W}}$,$M_{\mathbf{z}}$ and $M_{\mathbf{a}}$ such that $\|\mathbf{W}^k\| \leq M_{\mathbf{W}}$, $\|\mathbf{z}^k\| \leq M_{\mathbf{z}}$ and $\|\mathbf{a}^k\| \leq M_{\mathbf{a}}$.

This lemma ensures that the objective and all variables are bounded in the proposed mDLAM algorithm. Moreover, the gap between the same variables in the neighboring iterations(e.g. $\mathbf{W}^{k+1}$ and $\mathbf{W}^k$) is convergent to 0.

Lemma 4 (Subgradient Bound)**.** In Algorithm 4, there exist $C_2 = \max(\rho M_{\mathbf{a}}, \rho M_{\mathbf{a}}^2 + \theta_1^{k+1}, \rho M_{\mathbf{a}}^2 + \theta_2^{k+1}, \cdots, \rho M_{\mathbf{a}}^2 + \theta_L^{k+1})$, and $g_1^{k+1} \in \partial_{\mathbf{W}^{k+1}} F$ such that for any $k \in \mathbb{N}$

$$\|g_1^{k+1}\| \leq C_2(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{W}^k - \mathbf{W}^{k-1}\|).$$

The above lemma states that the subgradient of the objective is bounded by its variables. This suggests that the subgradient is convergent to 0, and thus proves its convergence to a stationary point.

3.4.2 Convergence of the Proposed mDLAM Algorithm

Next we discuss the convergence of the proposed mDLAM algorithm. The first theorem guarantees that the proposed mDLAM algorithm converges to a stationary point.

Theorem 4 (Convergence to a Stationary Point)**.** In Algorithm 4, for $\mathbf{W}$ in Problem 3, for any $\rho > 0$ and $\varepsilon > 0$, starting from any $\mathbf{W}^0$, any limit point $\mathbf{W}^*$ is a stationary point of Problem 3. That is, $0 \in \partial_{\mathbf{W}^*} F$.

As stated in Theorem 4, the convergence always holds no matter how $\mathbf{W}$ is initialized, and whatever hyperparameters ρ and ε are chosen. It is better than the proposed dlADMM algorithm in Chapter 2, which requires the hyperparameter to be sufficiently large.

Theorem 5 (Linear Convergence Rate)**.** In Algorithm 4, if F is locally strongly convex, then for any ρ, there exist $\varepsilon > 0$, $k_1 \in \mathbb{N}$ and $0 < C_1 < 1$ such that it holds for $k > k_1$ that

$$F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^* \leq C_1(F(\mathbf{W}^{k-1}, \mathbf{z}^{k-1}, \mathbf{a}^{k-1}) - F^*).$$

Theorem 5 shows that the proposed mDLAM algorithm converges linearly for sufficiently large iterations. Common loss functions like the square loss or the cross-entropy loss are locally strongly convex [115], which make F locally strongly convex. Therefore, Theorem 5 covers a wide range of loss functions. Compared with existing AM methods(e.g. dlADMM in Chapter 2) with a sublinear $o(1/k)$ convergence rate, the proposed mDLAM algorithm achieves a theoretically better linear convergence rate.

3.4.3 Discussion

We discuss convergence conditions of the proposed mDLAM algorithm compared with GD-type methods and the proposed dlADMM algorithm in Chapter 2. The comparison demonstrates that our convergence conditions are more general than others.

1. mDLAM versus GD

One influential work by Ghadimi et al. [30] guaranteed that the GD converges to a stationary point, which is similar to our convergence results. While the GD requires the objective function to be Lipschitz differentiable, bounded from below [30], our mDLAM allows for non-smooth functions such as ReLU. Therefore, our convergence conditions are milder than GD.

2. mDLAM versus dlADMM

Assumptions of our proposed mDLAM are milder than those of the proposed dlADMM in Chapter 2: the mDLAM requires activation functions to be quasilinear, which includes sigmoid, tanh, ReLU, and leaky ReLU, while the proposed dlADMM assumes that activation functions make subproblems solvable, which only includes ReLU and leaky ReLU. The such difference originates from different ways of addressing nonlinear activations: the proposed dlADMM treats them as L_2 penalties. For tanh and sigmoid, subproblems are difficult to solve and may refer to lookup tables [101]. However, the mDLAM relaxes them via inequality constraints, and subprob-

Dataset	Node#	Edge#	Class#	Feature#
Cora	2708	5429	7	1433
Pubmed	19717	44338	3	500
Citeseer	3327	4732	6	3703
Coauthor CS	18333	81894	15	6805

Table 3.1: Statistics of four benchmark datasets.

Method	Hyper-parameters	Cora	Pubmed	Citeseer	Coauthor CS
mDLAM	ρ	1×10^{-3}	0.01	5×10^{-3}	1×10^{-4}
GD	α	0.01	0.01	0.01	5×10^{-3}
Adadelta	α	0.01	0.1	0.01	0.05
Adagrad	α	5×10^{-3}	5×10^{-3}	0.01	5×10^{-3}
Adam	α	1×10^{-3}	5×10^{-4}	1×10^{-3}	1×10^{-3}
dlADMM	ρ	1×10^{-6}	1×10^{-6}	1×10^{-6}	1×10^{-6}

Table 3.2: Hyperparameter settings on four benchmark datasets.

lems have closed-form solutions.

3.5 Experiments

In this section, we evaluate the proposed mDLAM algorithm on four benchmark datasets. Convergence and efficiency are demonstrated. The performance of the proposed mDLAM algorithm is compared with several state-of-the-art optimizers. All experiments were conducted on a 64-bit machine with Intel(R) Xeon(R) Silver 4110 CPU and 64GB RAM.

3.5.1 Datasets and Parameter Settings

An important application of the MLP model is node classification on a graph based on augmented node features [15]. Specifically, given an adjacency matrix A and a node feature matrix H of a graph, we let the k-th augmented feature $X^k = HA^k (k =$

$0, 1, \cdots, 4$), which encodes information of graph topology via A^k, and then concatenate them into the input $X = [X_0, \cdots, X_4]$ [15]. The MLP model is used to predict the node class based on the input X. We set up an architecture of three layers, each of which has 100 neurons. The activation function was set to ReLU. The number of epochs was set to 200. We test our model on four benchmark datasets: Cora [86], Pubmed [86], Citeseer [86] and Coauthor CS [89], whose statistics are shown in Table 3.1.

Gradient Descent(GD) [4], Adaptive learning rate method(Adadelta) [118], Adaptive gradient algorithm(Adagrad) [28], Adaptive momentum estimation(Adam) [52], and the proposed dlADMM in Chapter 2 are state-of-the-art methods and hence were served as comparison methods. The full batch dataset was used for training models. All parameters were chosen by maximizing the accuracy of training datasets. Table 3.2 shows hyperparameters of all methods: for the proposed mDLAM algorithm, ρ controls quadratic terms in Problem 3; α is a learning rate in the comparison methods except for the proposed dlADMM. ρ controls a linear constraint in the proposed dlADMM algorithm. The other hyperparameter ε is chosen adaptively as follows: $\varepsilon^{k+1} = \max(\varepsilon^k/2, 0.001)$ with $\varepsilon^0 = 100$. This makes inequality constraints relaxed at the early stage(i.e. ε^k is large and hence constraints are easy to satisfy) and then tightens them as the mDLAM iterates.

3.5.2 Convergence

Firstly, we investigate the convergence of the proposed mDLAM algorithm on four benchmark datasets using the hyperparameters summarized in Table 3.2. The relationship between the objective and the number of epochs is shown in Figure 3.1. Overall, the objectives on the four datasets all decrease monotonically, which demonstrates the convergence of the proposed mDLAM algorithm. Nevertheless, objective curves vary in tendency: the curves on the Cora and Pubmed datasets drop drasti-

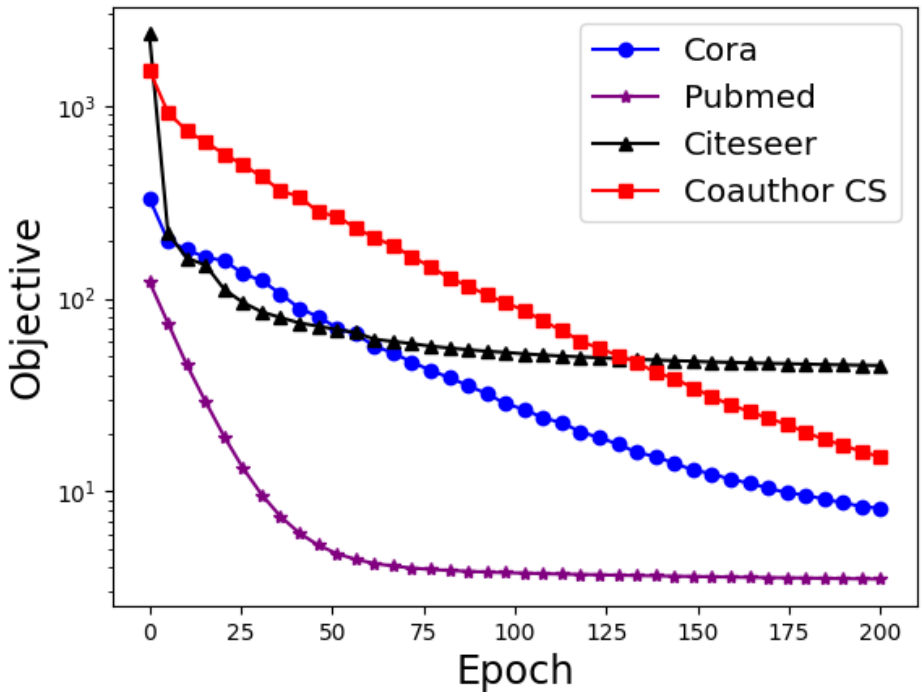
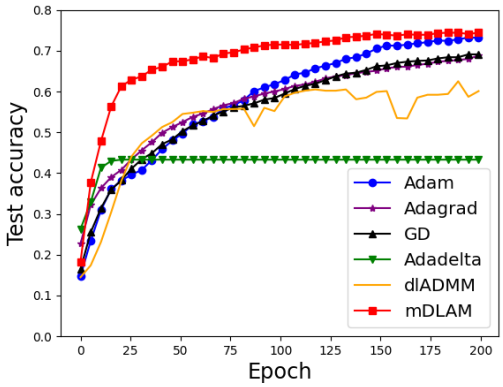


Figure 3.1: Convergence curves of the proposed mDLAM algorithm on four datasets.

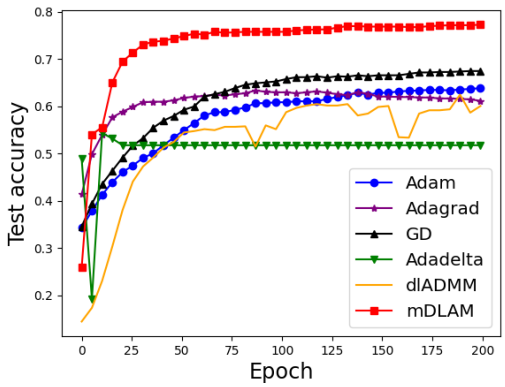
cally at the beginning and then reach the plateau when the epoch is around 75, while the curves on the other two datasets keep a downward tendency for the entire 200 epochs. Moreover, the objective on the Pubmed dataset is the lowest at the end of the training, while the objective on the Citeseer dataset is in the vicinity of 80, at least 60% higher than the objectives on the remaining datasets. It is easy to observe that all curves decline linearly when the epoch is higher than 100. This validates the linear convergence rate of our proposed mDLAM algorithm(i.e. Theorem 5).

3.5.3 Performance

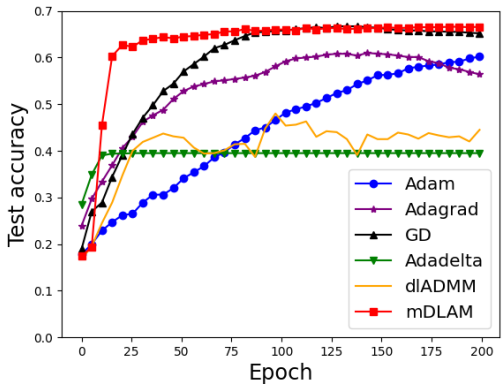
Next, the performance of the proposed mDLAM algorithm is compared against five state-of-the-art methods, as is illustrated in Figure 3.2. The X-axis and Y-axis represent epoch and test accuracy, respectively. Overall, the proposed mDLAM algorithm is superior to all other algorithms on four datasets, which has not only the highest test



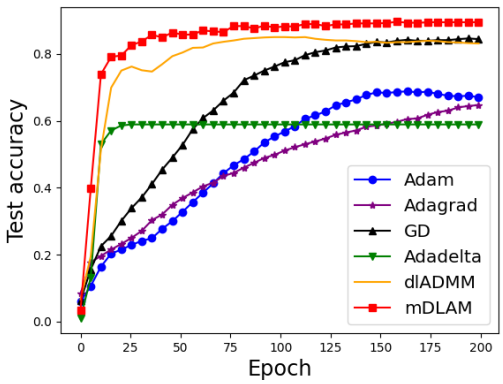
(a). Cora.



(b). Pubmed.

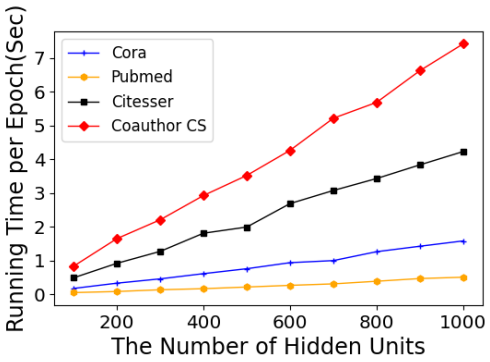


(c). Citeseer.

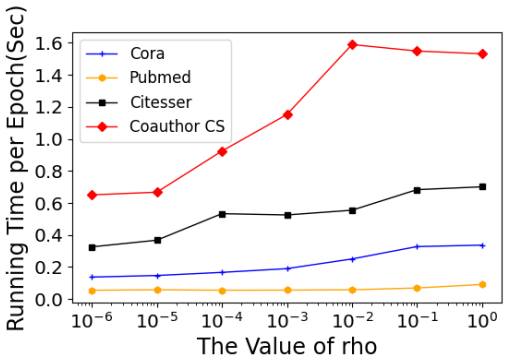


(d). Coauthor CS.

Figure 3.2: Test accuracy of the proposed mDLAM algorithm against all comparison methods.



(a). Running time versus the number of neurons.



(b). Running time versus the value of ρ.

Figure 3.3: The relationship between the running time of the proposed mDLAM algorithm and:(a) the number of neurons;(b) the value of ρ.

accuracy but also the fastest convergence speed. For example, the proposed mDLAM achieves 70% test accuracy on the Cora dataset when the epoch is 100, while GD only attains 60%, and the Adadelta reaches the plateau of around 40%; As another example, the test accuracy of the proposed mDLAM on the Coauthor CS dataset is over 80% at the 25-th epoch, whereas most comparison methods such as Adam and GD reach half of its accuracy(i.e. 40%). The Adadelta algorithm performs the worst among all comparison methods: it converges to a low test accuracy at the early stage, which is usually half of the accuracy accomplished by the proposed mDLAM algorithm. The other four comparison methods except Adagrad are on par with mDLAM in some cases: for example, the curves of dlADMM and GD are marginally behind that of mDLAM on the Coauthor CS dataset, and the performance of Adam almost reaches that of mDLAM on the Cora dataset. It is interesting to observe that curves of some methods decline at the end of 200 epochs such as the Adagrad on the Pubmed dataset and the Adam on the Coauthor CS dataset.

3.5.4 Sensitivity Analysis

We explore concerning factors of the running time and the test accuracy in this section.

Running Time

Moreover, it is important to explore the running time of the proposed mDLAM concerning two factors: the number of neurons and the value of ρ. The running time was averaged by 200 epochs. Figure 3.3(a) depicts the relationship between the running time and the number of neurons on four datasets, where the number of neurons ranges from 100 to $1,000$. The running times on all datasets are below 1 second per epoch when the number of neurons is 100, and increase linearly with the number of neurons in general. However, the rates of increase vary on different datasets: the curve on the Coauthor CS dataset has the sharpest slope, which reaches seven seconds per epoch when 1000 neurons are applied, while the curve on the Pubmed dataset climbs slowly, which never surpasses 1 second. The curves on the Cora and the Citeseer datasets demonstrate a steady increase.

To investigate the relationship between the running time per epoch and the value of ρ, we change ρ from 10^{-6} to 1 while fixing others. Similar to Figure 3.3(a), the running time per epoch demonstrates a linear increase concerning the value of ρ in general, as shown in Figure 3.3(b). Specifically, the curve on the Coauthor CS dataset is still the highest in slope, whereas the slope on the Pubmed dataset is the lowest. Moreover, the effect of the value of ρ is less obvious than the number of neurons. For example, in Figure 3.3(b) when ρ is enlarged from 10^{-6} to 10^{-2}, the running time on the Coauthor CS dataset merely ascends from around 0.65 to 1.6, while the increment of the running time on other datasets is less than 0.2. Moreover, a larger ρ may reduce the running time. For instance, when ρ increases from 10^{-2} to 1, the running time on the Coauthor CS dataset drops slightly from 1.6 seconds to 1.5 seconds per

Cora					
Epoch	40	80	120	160	200
$\rho = 1 \times 10^{-4}$	0.677	0.695	0.695	0.693	0.692
$\rho = 1 \times 10^{-3}$	0.664	0.701	0.721	0.737	0.742
$\rho = 1 \times 10^{-2}$	0.562	0.581	0.604	0.623	0.638
Pubmed					
Epoch	40	80	120	160	200
$\rho = 1 \times 10^{-4}$	0.471	0.407	0.407	0.407	0.407
$\rho = 1 \times 10^{-3}$	0.663	0.645	0.640	0.650	0.649
$\rho = 1 \times 10^{-2}$	0.743	0.758	0.762	0.768	0.773
Citeseer					
Epoch	40	80	120	160	200
$\rho = 1 \times 10^{-4}$	0.528	0.529	0.530	0.531	0.535
$\rho = 1 \times 10^{-3}$	0.651	0.665	0.664	0.664	0.666
$\rho = 1 \times 10^{-2}$	0.631	0.638	0.642	0.648	0.653
Coauthor CS					
Epoch	40	80	120	160	200
$\rho = 1 \times 10^{-4}$	0.843	0.881	0.888	0.896	0.894
$\rho = 1 \times 10^{-3}$	0.780	0.807	0.825	0.839	0.835
$\rho = 1 \times 10^{-2}$	0.688	0.719	0.724	0.737	0.738

Table 3.3: The effect of ρ on the test accuracy of the proposed mDLAM algorithm on four datasets.

epoch. The running times on the Cora and the Citeseer datasets climb steadily.

Test Accuracy

Finally, we investigate the effects of hyperparameters on test accuracy, namely, the value of ρ and ε. Because ε is dynamically set, we test its initial value ε^0. Table 3.3 demonstrates the relationship between test accuracy and ρ on four datasets. ρ was chosen from $\{1 \times 10^{-4}, 1 \times 10^{-3}, 1 \times 10^{-2}\}$. Overall, the choice of ρ has a significant effect on the test accuracy. For example, when ρ is changed from 1×10^{-4} to 1×10^{-3} on the Pubmed dataset, the performance has improved by approximately 60%, and the gain of performance is even roughly 90% if it is modified to 1×10^{-2}. On other datasets, the change of ρ affects test accuracy by around 20%. For instance, the test accuracy on the Cora dataset and the Coauthor CS dataset can be improved to 0.74

Cora					
Epoch	40	80	120	160	200
$\varepsilon^0 = 1$	0.620	0.679	0.712	0.735	0.743
$\varepsilon^0 = 10$	0.646	0.689	0.718	0.741	0.741
$\varepsilon^0 = 100$	0.664	0.701	0.721	0.737	0.742
Pubmed					
Epoch	40	80	120	160	200
$\varepsilon^0 = 1$	0.717	0.744	0.756	0.759	0.763
$\varepsilon^0 = 10$	0.731	0.753	0.759	0.762	0.765
$\varepsilon^0 = 100$	0.743	0.758	0.762	0.768	0.773
Citeseer					
Epoch	40	80	120	160	200
$\varepsilon^0 = 1$	0.564	0.615	0.638	0.653	0.663
$\varepsilon^0 = 10$	0.584	0.626	0.643	0.657	0.662
$\varepsilon^0 = 100$	0.640	0.656	0.664	0.663	0.668
Coauthor CS					
Epoch	40	80	120	160	200
$\varepsilon^0 = 1$	0.834	0.875	0.887	0.893	0.894
$\varepsilon^0 = 10$	0.852	0.866	0.892	0.893	0.893
$\varepsilon^0 = 100$	0.843	0.881	0.888	0.896	0.894

Table 3.4: The effect of the initial value of ε(i.e. ε^0) on the test accuracy of the proposed mDLAM algorithm on four datasets.

and 0.89 if we set $\rho = 1 \times 10^{-3}$ and $\rho = 1 \times 10^{-4}$, respectively. The test accuracy on the Citeseer dataset is relatively robust to the change of ρ. As ρ varies from 1×10^{-3} to 1×10^{-2}, the test accuracy remains stable. The test accuracy generally increases as the proposed mDLAM algorithm iterates. However, there are some exceptions: for example, the test accuracy has dropped slightly from 0.66 to 0.65 when $\rho = 1 \times 10^{-3}$ on the Pubmed dataset. Table 3.4 shows the relationship between test accuracy and the initial value of ε(i.e. ε^0) on four datasets. ε^0 was chosen from $\{1, 10, 100\}$. The test accuracy is resistant to the change of ε^0. For example, the test accuracy on the Coauthor CS dataset is in the vicinity of 0.89 no matter which ε is chosen. Moreover, the larger a ε^0 is the faster convergence speed the proposed mDLAM algorithm gains. For instance, when $\varepsilon = 100$, the test accuracy is 0.08 better than that in the case where $\varepsilon = 1$ on the Citeseer dataset. Compared with Tables 3.3 and 3.4, the effect of

ρ is more significant than that of ε^0.

3.6 Conclusion

In this chapter, we propose a novel formulation of the original neural network problem and a novel monotonous Deep Learning Alternating Minimization(mDLAM) algorithm. Specifically, the nonlinear constraint is projected into a convex set so that all subproblems are solvable. The Nesterov acceleration technique is applied to boost the convergence of the proposed mDLAM algorithm. Furthermore, a mild assumption is established to prove the convergence of our mDLAM algorithm. Our mDLAM algorithm can achieve a linear convergence rate, which is theoretically better than existing AM methods. The effectiveness of the proposed mDLAM algorithm is demonstrated via the outstanding performance on four benchmark datasets compared with state-of-the-art optimizers.

Chapter 4

The pdADMM Algorithm

4.1 Introduction

Due to wide applications and significant success in various applications, the training of Deep Neural Network(DNN) models has gained ever-increasing attention from the machine learning community. In recent years, the constant improvement of DNNs' performance is accompanied by a fast increase in models' complexity and size, which indicates a clear trend toward larger and deeper networks. Such a trend leads to severe challenges for large models to fit into a single computing unit(e.g., GPU), and raises urgent demands for partitioning the model into different computing devices to parallel training. However, the inherent bottleneck from GD which prevents the gradients of different layers from being calculated in parallel. This is because in GD gradient calculations of one layer tightly depend on and have to wait for the calculated results of all previous layers, which prevents the gradients of different layers from being calculated in parallel.

To work around the drawback of gradient-based methods, alternating minimization methods have caught fast increasing attention in recent years. Amongst them for deep learning optimization, ADMM-based methods are deemed to have great po-

tential for parallelism of deep neural network training, due to their inherent nature, which can break an objective into multiple subproblems, each of which can be solved in parallel [6].

Despite the potential, a parallel algorithm based on ADMM for deep neural network training has rarely been explored, developed, and evaluated until now, due to the layer dependency among subproblems of ADMM. Even though the ADMM reduces the layer dependency compared with GD, one subproblem of ADMM is dependent on its previous subproblem. Therefore, existing ADMM-based optimizers still update parameters sequentially.

To handle the difficulties of layer dependency, in this chapter we propose a novel parallel deep learning Alternating Direction Method of Multipliers(pdADMM) optimization framework to train large-scale neural networks. Our contributions in this chapter include:

- We propose a novel reformulation of the Multi-Layer Perceptron(MLP) neural network problem, which splits a neural network into independent layer partitions and allows for ADMM to achieve model parallelism.

- We present a model-parallelism version of the ADMM algorithm to train an MLP neural network. All parameters in each layer can be updated in parallel to speed up the training process significantly. All subproblems generated by the pdADMM algorithm are discussed in detail.

- We investigated the convergence properties of parallel ADMM in the common nonlinear activation functions such as the Rectified linear unit(ReLU), and we prove that the pdADMM converges to a state-of-the-art stationary point with a sublinear convergence rate $o(1/k)$.

- We conduct extensive experiments on six benchmark datasets to show the massive speedup of the proposed pdADMM as well as its competitive performance

with state-of-the-art optimizers.

The organization of this chapter is shown as follows: In Section 4.2, we summarize recent related research work to this chapter. In Section 4.3, we formulate the novel pdADMM algorithm to train a MLP neural network. In Section 4.4, the convergence guarantee of pdADMM to a stationary point is provided. Extensive experiments on benchmark datasets to demonstrate the convergence, speedup and comparable performance of pdADMM are shown in Section 4.5, and Section 4.6 concludes this work.

4.2 Related Work

Distributed ADMM: ADMM is one of the commonly applied techniques in distributed optimization. Overall, the previous works on distributed ADMM can be classified into two categories: synchronous problems and asynchronous problems. Synchronous problems usually require workers to optimize parameters in time before the master update the consensus variable, while asynchronous problems allow some workers to delay parameter updates. Most literature focused on the application of the distributed ADMM to synchronous problems. For example, Mota et al. utilized the distributed ADMM for the congestion control problem [70]; Makhdoumi and Ozdaglar studied the convergence properties of the distributed ADMM on the network communication problem. For more work, please refer to [10, 11, 90, 116, 126]. On the other hand, a handful of papers investigated how asynchronous problems can be addressed by distributed ADMM. For instance, Zhang et al., Wei et al., Chang et al and Hong proved the convergence of the distributed ADMM on asynchronous problems [12, 13, 39, 123, 108]. Kumar et al. discussed the application of the ADMM on multi-agent problems over heterogeneous networks [56]. However, there still lacks a general framework for ADMM to train deep neural networks in a distributed fash-

ion.

Distributed Deep Learning: With the increased volume of data and layers of neural networks, there is a need to design distributed systems to train a deep neural network for large-scale applications. Most recent papers have proposed gradient-based distributed systems to train neural networks: For example, Wen et al. proposed Terngrad to accelerate distributed deep learning in data parallelism [109]; Sergeev et al. presented an open-source library Horovod to reduce communication overhead [88]. Other systems include SINGA [77], Mxnet[16], TicTac [36] and Poseidon [122].

Data and Model Parallelism: Data parallelism focuses on distributing data across different processors, which can be implemented in parallel. Scaling GD is one of the most common ways to reach data parallelism [129]. For example, the distributed architecture, Poseidon, is achieved by scaling GD through overlapping communication and computation over networks. The recently proposed ADMM [93, 101] is another way of data parallelism: each subproblem generated by ADMM can be solved in parallel. However, data parallelism suffers from the bottleneck of a neural network: for GD, the gradient should be transmitted through all processors; for ADMM, the parameters in one layer are subject to these in its previous layer. As a result, this leads to heavy communication costs and time delays. Model parallelism, however, can solve this challenge because model parallelism splits a neural network into many independent partitions. In this way, each partition can be optimized in parallel and hence reduce time delay. For instance, Parpas and Muir proposed a parallel-in-time method from the perspective of dynamic systems [79]; Huo et al. introduced a feature replay algorithm to achieve model parallelism [43]. Zhuang et al. broke layer dependency by introducing the delayed gradient [127]. However, to the best of our knowledge, there still lacks an exploration of how to achieve model parallelism via ADMM.

4.3 pdADMM

We propose the pdADMM algorithm in this section. Specifically, Section 4.3.1 introduces the existing deep learning ADMM method, reformulates the problem, and presents the pdADMM algorithm in detail. Section 4.3.2 discusses all subproblems generated by pdADMM and the strategy to train a large-scale deep neural network via pdADMM.

4.3.1 Problem Reformulation

Problem 1 has been addressed by the proposed dlADMM and mDLAM in Chapters 2 and 3, respectively. However, parameters in one layer are dependent on its neighboring layers and hence can not achieve parallelism. For example, the update of a_{l+1} on the $(l+1)$-th layer needs to wait before z_l on the l-th layer is updated. In order to address layer dependency, we relax Problem 1 to Problem 4 as follows:

Problem 4.

$$\min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}} F(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}) = R(z_L,y)+(\nu/2)(\sum_{l=1}^{L}\|z_l - W_l p_l\|_2^2 + \sum_{l=1}^{L-1}\|q_l - f_l(z_l)\|_2^2)$$
$$s.t.\ p_{l+1} = q_l.$$

where p_l and q_l are the input and the output of the i-th layer, respectively, $\mathbf{p} = \{p_l\}_{l=1}^{L}$, $\mathbf{W} = \{W_l\}_{l=1}^{L}$, $\mathbf{z} = \{z_l\}_{l=1}^{L}$, $\mathbf{q} = \{q_l\}_{l=1}^{L-1}$, and $\nu > 0$ is a tuning parameter. We reduce layer dependency by splitting the output of the l-th layer and the input of the $(l+1)$-th layer into two variables p_{l+1} and q_l, respectively. As $\nu \to \infty$, Problem 4 approaches Problem 1.

The high-level overview of the pdADMM algorithm is shown in Figure 4.1. Specifically, by breaking the whole neural network into multiple layers, each of which can be optimized by an independent worker. Therefore, layerwise training can be imple-

mented in parallel. Moreover, the gradient vanishing problem can be avoided in this way. This is because the accumulated gradient calculated by the backpropagation algorithm is split into layerwise components.

Now we follow the ADMM routine to solve Problem 4, the augmented Lagrangian function is formulated mathematically as follows:

$$\begin{aligned}
&L_\rho(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}, \mathbf{u}) \\
&= F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}) + \sum\nolimits_{l=1}^{L-1} (u_l^T(p_{l+1} - q_l) + (\rho/2)\|p_{l+1} - q_l\|_2^2) \\
&= R(z_L, y) + \phi(p_1, W_1, z_1) + \sum\nolimits_{l=2}^{L} \phi(p_l, W_l, z_l, q_{l-1}, u_{l-1}) \\
&+ (\nu/2) \sum\nolimits_{l=1}^{L-1} \|q_l - f_l(z_l)\|_2^2.
\end{aligned}$$

where $\phi(p_1, W_1, z_1) = (\nu/2)\|z_1 - W_1 p_1\|_2^2$, $\phi(p_l, W_l, z_l, q_{l-1}, u_{l-1}) = (\nu/2)\|z_l - W_l p_l\|_2^2 + u_{l-1}^T(p_l - q_{l-1}) + (\rho/2)\|p_l - q_{l-1}\|_2^2$, $u_l (l = 1, \cdots, L-1)$ are dual variables, $\rho > 0$ is a parameter, and $\mathbf{u} = \{u_l\}_{l=1}^{L-1}$. The detail of the pdADMM is shown in Algorithm 5. Specifically, Lines 5-8 update primal variables $\mathbf{p}$, $\mathbf{W}$, $\mathbf{z}$ and $\mathbf{q}$, respectively, while Line 11 updates the dual variable $\mathbf{u}$. the discussion on how to solve subproblems generated by pdADMM is detailed in the next section.

Algorithm 5: The proposed pdADMM algorithm

Require: y, $p_1 = x$, ρ, ν.
Ensure: $\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}$.
Initialize $k = 0$.
while $\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k$ not converged **do**
Update p_l^{k+1} of different l by Equation (4.1) in parallel.
Update W_l^{k+1} of different l by Equation (4.2) in parallel.
Update z_l^{k+1} of different l by Equations (4.3) and (4.4) in parallel.
Update q_l^{k+1} of different l by Equation (4.5) in parallel.
$r_l^k \leftarrow p_{l+1}^{k+1} - q_l^{k+1} (l = 1, \cdots, L)$ in parallel # Compute residuals.
Update u_l^{k+1} of different l by Equation (4.6) in parallel.
$k \leftarrow k + 1$.
end while
Output $\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}$.

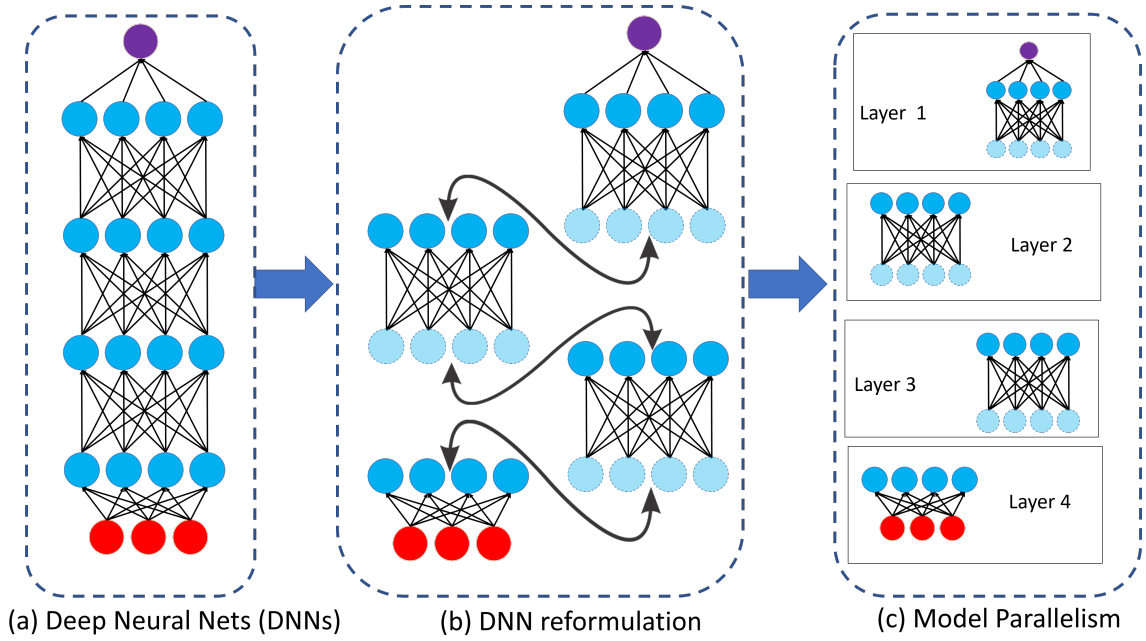


Figure 4.1: The overview of the proposed pdADMM algorithm.

4.3.2 Solutions to All Subproblems

In this section, we discuss how to solve all subproblems generated by pdADMM in detail.

1. Update $\mathbf{p}^{k+1}$

The variable $\mathbf{p}^{k+1}$ is updated as follows:

$$p_l^{k+1} \leftarrow \arg\min_{p_l} L_\rho(\mathbf{p}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) = \phi(p_l, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)$$

Because W_l and p_l are coupled in ϕ, solving p_l should require the time-consuming operation of matrix inversion of W_l. To handle this, we apply similar quadratic approximation techniques as used in dlADMM [101] as follows:

$$p_l^{k+1} \leftarrow \arg\min_{p_l} U_l(p_l; \tau_l^{k+1}) \tag{4.1}$$

where $U_l(p_l;\tau_l^{k+1}) = \phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + (\nabla_{p_l^k}\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)(p_l - p_l^k) + (\tau_l^{k+1}/2)\|p_l - p_l^k\|_2^2$, and $\tau_l^{k+1} > 0$ is a parameter. τ_l^{k+1} should satisfy $\phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) \leq U_l(p_l^{k+1};\tau_l^{k+1})$. The solution to Equation (4.1) is: $p_l^{k+1} \leftarrow p_l^k - \nabla_{p_l^k}\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)/\tau_l^{k+1}$.

2. Update $\mathbf{W}^{k+1}$

The variable $\mathbf{W}^{k+1}$ is updated as follows:

$$\begin{aligned} W_l^{k+1} &\leftarrow \arg\min_{W_l} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\ &= \arg\min_{W_l} \begin{cases} \phi(p_1^{k+1}, W_1, z_1^k) & l = 1 \\ \phi(p_l^{k+1}, W_l, z_l^k, q_{l-1}^k, u_{l-1}^k) & 1 < l \leq L \end{cases} \end{aligned}$$

Similar to updating p_l, the following subproblem should be solved instead:

$$W_l^{k+1} \leftarrow \arg\min_{W_l} V_l(W_l;\theta_l^{k+1}) \tag{4.2}$$

where

$$\begin{aligned} V_1(W_1;\theta_1^{k+1}) &= \phi(p_1^{k+1}, W_1^k, z_1^k) + \nabla_{W_1^k}\phi^T(p_1^{k+1}, W_1^k, z_1^k)(W_1 - W_1^k) \\ &\quad + (\theta_l^{k+1}/2)\|W_1 - W_1^k\|_2^2 \\ V_l(W_l;\theta_l^{k+1}) &= \phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + \nabla_{W_l^k}\phi^T(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)(W_l - W_l^k) \\ &\quad + (\theta_l^{k+1}/2)\|W_l - W_l^k\|_2^2 \end{aligned}$$

and θ_l^{k+1} is a parameter, which should satisfy $\phi(p_1^{k+1}, W_1^{k+1}, z_1^k) \leq V(W_1^{k+1};\theta_1^{k+1})$ and $\phi(p_l^{k+1}, W_l^{k+1}, z_l^k, q_{l-1}^k, u_{l-1}^k) \leq V(W_l^{k+1};\theta_l^{k+1})(1 < l < L)$. The solution to Equation

(4.2) is shown as follows:

$$W_l^{k+1} \leftarrow W_l^k - \begin{cases} \nabla_{W_1^k}\phi(p_1^{k+1}, W_1^k, z_1^k)/\theta_l^{k+1} & l = 1 \\ \nabla_{W_l^k}\phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)/\theta_l^{k+1} & 1 < l \leq L \end{cases}$$

3. Update $\mathbf{z}^{k+1}$

The variable $\mathbf{z}^{k+1}$ is updated as follows:

$$z_l^{k+1} \leftarrow \arg\min_{z_l}(\nu/2)\|z_l - W_l^{k+1}p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l)\|_2^2 + (\nu/2)\|z_l - z_l^k\|_2^2$$

$$(l < L) \tag{4.3}$$

$$z_L^{k+1} \leftarrow \arg\min_{z_l} R(z_L, y) + (\nu/2)\|z_L - W_L^{k+1}p_L^{k+1}\|_2^2 \tag{4.4}$$

where a quadratic term $(\nu/2)\|z_l - z_l^k\|_2^2$ is added in Equation (4.3) to control z_l^{k+1} to close to z_l^k. Equation (4.4) is convex, which can be solved by Fast Iterative Soft Thresholding Algorithm(FISTA) [2].

For Equation (4.3), nonsmooth activations usually lead to closed-form solutions [101, 99]. For example, for ReLU $f_l(z_l) = \max(z_l, 0)$, the solution to Equation (4.3) is shown as follows:

$$z_l^{k+1} = \begin{cases} \min((W_l^{k+1}p_l^{k+1} + z_l^k)/2, 0) & z_l^{k+1} \leq 0. \\ \max((W_l^{k+1}p_l^{k+1} + q_l^k + z_l^k)/3, 0) & z_l^{k+1} \geq 0. \end{cases}$$

For smooth activations such as tanh and sigmoid, a lookup-table is recommended [101].

4. Update $\mathbf{q}^{k+1}$

The variable $\mathbf{q}^{k+1}$ is updated as follows:

$$q_l^{k+1} \leftarrow \arg\min_{q_l} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}, \mathbf{u}^k) = \arg\min_{q_l} \phi(p_{l+1}^{k+1}, W_{l+1}^{k+1}, z_{l+1}^{k+1}, q_l, u_l^k). \tag{4.5}$$

Equation (4.5) has a closed-form solution as follows:

$$q_l^{k+1} \leftarrow (\rho p_{l+1}^{k+1} + u_l^k + \nu f_l(z_l^{k+1}))/(\rho + \nu)$$

5. Update $\mathbf{u}^{k+1}$

The variable $\mathbf{u}^{k+1}$ is updated as follows:

$$u_l^{k+1} \leftarrow u_l^k + \rho(p_{l+1}^{k+1} - q_l^{k+1}) \tag{4.6}$$

Finally, Our proposed pdADMM can be efficient for training a deep MLP neural network. To achieve this, we begin by training a swallow neural network with the first few layers of the deep neural network, then more layers are added for training step by step until finally all layers are involved in the training process. The pdADMM can achieve good performance as well as reduce training costs with this strategy.

4.4 Convergence Analysis

In this section, the theoretical convergence of the proposed pdADMM algorithm. Firstly, the Lipschitz continuity and coercivity are defined as follows:

Definition 1. (Lipschitz Continuity) A function $g(x)$ is Lipschitz continuous if there

exists a constant $D > 0$ such that $\forall x_1, x_2$, the following holds

$$\|g(x_1) - g(x_2)\| \leq D\|x_1 - x_2\|.$$

Definition 2. (Coercivity) A function $h(x)$ is coerce over the feasible set $\mathscr{F}$ means that $h(x) \to \infty$ if $x \in \mathscr{F}$ and $\|x\| \to \infty$.

Then the following assumption is required for convergence analysis.

Assumption 5. $f_l(z_l)$ is Lipschitz continuous with coefficient $S > 0$, and F is coercive. Moreover, $\partial f_l(z_l)$ is bounded, i.e. there exists $M > 0$ such that $\|\partial f_l(z_l)\| \leq M$.

Assumption 5 is mild to satisfy: most common activation functions such as ReLU and leaky ReLU satisfy Assumption 5. No assumption is needed on the risk function $R(z_L, y)$, which shows that the convergence condition of our proposed pdADMM is milder than that of the dlADMM, which requires $R(z_L, y)$ to be Lipschitz differentiable [101]. Due to space limit, the detailed proofs are provided in Section C.1 and Section C.2 in the Appendix. The technical proofs follow a similar routine as dlADMM [101]. The difference consists in the fact that the dual variable u_l is controlled by q_l and z_l(Lemma 21 in Section C.1 in the Appendix), which holds under Assumption 5, while u_l can be controlled only by z_l in the convergence proof of dlADMM. The first lemma shows that the objective keeps decreasing when ρ is sufficiently large.

Lemma 5 (Decreasing Objective)**.** If $\rho > \max(4\nu S^2, (\sqrt{17} + 1)\nu/2)$, there exist $C_1 = \nu/2 - 2\nu^2 S^2/\rho > 0$ and $C_2 = \rho/2 - 2\nu^2/\rho - \nu/2 > 0$ such that it holds for any

$k \in \mathbb{N}$ that

$$
\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \sum_{l=2}^{L} (\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 + \sum_{l=1}^{L} (\theta_l^{k+1}/2)\|W_l^{k+1} - W_l^k\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_1\|z_l^{k+1} - z_l^k\|_2^2 + (\nu/2)\|z_L^{k+1} - z_L^k\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{k+1} - q_l^k\|_2^2 \qquad (4.7)
\end{aligned}
$$

The second Lemma illustrates that the objective is bounded from below when ρ is large enough, and all variables are bounded.

Lemma 6 (Bounded Objective)**.** If $\rho > \nu$, then $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is lower bounded. Moreover, $\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k$,and $\mathbf{u}^k$ are bounded, i.e. there exist $\mathbb{N}_\mathbf{p}$, $\mathbb{N}_\mathbf{W}$, $\mathbb{N}_\mathbf{z}$, $\mathbb{N}_\mathbf{q}$, and $\mathbb{N}_\mathbf{u} > 0$, such that $\|\mathbf{p}^k\| \leq \mathbb{N}_\mathbf{p}$, $\|\mathbf{W}^k\| \leq \mathbb{N}_\mathbf{W}$, $\|\mathbf{z}^k\| \leq \mathbb{N}_\mathbf{z}$, $\|\mathbf{q}^k\| \leq \mathbb{N}_\mathbf{q}$, and $\|\mathbf{u}^k\| \leq \mathbb{N}_\mathbf{u}$.

Based on Lemmas 5 and 6, the following theorem ensures that the objective is convergent.

Theorem 6 (Convergent Objective)**.** If $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, then $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent. Moreover, $\lim_{k\to\infty} \|\mathbf{p}^{k+1} - \mathbf{p}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{q}^{k+1} - \mathbf{q}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0$.

Proof. From Lemmas 5 and 6, we know that $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent because a monotone bounded sequence converges. Moreover, we take the limit on both sides of Inequality (4.7) to obtain

$$
\begin{aligned}
&0 = \lim_{k\to\infty} L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - \lim_{k\to\infty} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \lim_{k\to\infty} (\sum_{l=2}^{L} (\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 + \sum_{l=1}^{L} (\theta_l^{k+1}/2)\|W_l^{k+1} - W_l^k\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_1\|z_l^{k+1} - z_l^k\|_2^2 + (\nu/2)\|z_L^{k+1} - z_L^k\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{k+1} - q_l^k\|_2^2) \geq 0
\end{aligned}
$$

Because $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent, then $\lim_{k\to\infty} \|\mathbf{p}^{k+1} - \mathbf{p}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 = 0$, and $\lim_{k\to\infty} \|\mathbf{q}^{k+1} - \mathbf{q}^k\|_2^2 = 0$. $\lim_{k\to\infty} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0$ is derived from Lemma 21 in Section C.1 in the Appendix. □

The third lemma guarantees that the subgradient of the objective is upper bounded, which is stated as follows:

Lemma 7 (Bounded Subgradient)**.** There exists a constant $C > 0$ and $g^{k+1} \in \partial L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}\|g^{k+1}\| \leq C(&\|\mathbf{p}^{k+1} - \mathbf{p}^k\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| \\ &+ \|\mathbf{u}^{k+1} - \mathbf{u}^k\|)\end{aligned}$$

Now based on Theorem 6, and Lemma 7, the convergence of the pdADMM algorithm to a stationary point is presented in the following theorem.

Theorem 7 (Convergence to a stationary Point)**.** If $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, then for the variables $(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}, \mathbf{u})$ in Problem 4, starting from any initialization $(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$, and any limit point is a stationary point of Problem 4 (i.e. $0 \in \partial L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$). In other words,

$$\begin{aligned}&p_{l+1}^* = q_l^*,\ \nabla_{\mathbf{p}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0,\ \nabla_{\mathbf{W}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0, \\ &0 \in \partial_{\mathbf{z}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*),\ \nabla_{\mathbf{q}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0.\end{aligned}$$

Proof. From Lemma 6, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point$(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ because a bounded sequence has at least a limit point. From Lemma 7 and Theorem 6, $\|g^{k+1}\| \to 0$ as $k \to \infty$. According to the definition of general subgradient(Defintion

8.3 in [84]), we have $0 \in \partial L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$. In other words, every limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ is a stationary point. □

Theorem 7 shows that our proposed pdADMM algorithm converges for sufficiently large ρ, which is consistent with previous literature [101]. Next, the following theorem ensures the sublinear convergence rate $o(1/k)$ of the proposed pdADMM algorithm, whose proof is in the Appendix.

Theorem 8 (Convergence Rate)**.** For a sequence $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$, define $c_k = \min_{0\le i\le k}(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1} - p_l^i\|_2^2 + \sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1} - W_l^i\|_2^2 + \sum_{l=1}^{L-1} C_1\|z_l^{i+1} - z_l^i\|_2^2 + (\nu/2)\|z_L^{i+1} - z_L^i\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{i+1} - q_l^i\|_2^2)$ where $C_1 = \nu/2 - 2\nu^2S^2/\rho > 0$ and $C_2 = \rho/2 - 2\nu^2/\rho - \nu/2 > 0$, then the convergence rate of c_k is $o(1/k)$.

4.5 Experiments

In this section, we evaluate the performance of the proposed pdADMM using six benchmark datasets. Speedup, convergence, and accuracy performance are compared with several state-of-the-art optimizers. All experiments were conducted on a 64-bit machine with Intel Xeon(R) silver 4114 Processor and 48GB RAM.

4.5.1 Datasets

In this experiment, six benchmark datasets were used for performance evaluation:

1. MNIST [58]. The MNIST dataset has ten classes of handwritten-digit images, which was firstly introduced by Lecun et al. in 1998 [58]. It contains 55,000 training samples and 10,000 test samples with 196 features each, which is provided by the Keras library [17].

2. Fashion MNIST [112]. The Fashion MNIST dataset has ten classes of assortment images on the website of Zalando, which is Europe's largest online fashion platform [112]. The Fashion-MNIST dataset consists of 60,000 training samples and 10,000

test samples with 196 features each.

3. Kuzushiji-MNIST(kMNIST) [20]. The kMNIST dataset has ten classes, each of which is a character to represent each of the 10 rows of Hiragana. The kMNIST dataset consists of 60,000 training samples and 10,000 test samples with 196 features each.

4. Street View House Numbers(SVHN) [72]. The SVHN dataset is obtained from house numbers in Google Street View images. It consists of ten classes of digits. In our experiments, we use three classes '0', '1' and '2'. The number of training data and test data are 24,446 and 9,248, respectively, with 768 features each.

5. CIFAR10 [55]. CIFAR10 is a collection of color images with 10 different classes. In our experiments, we use two classes '0' and '6'. The number of training data and test data are 12,000 and 2,000, respectively, with 768 features each.

6. CIFAR100 [55]. CIFAR100 is similar to CIFAR10 except that CIFAR100 has 100 classes. In our experiments, we use two classes '0' and '2'. The number of training data and test data are 5,000 and 1,000, respectively, with 768 features each.

4.5.2 Speedup

MNIST dataset			
Neurons#	Serial pdADMM(sec)	pdADMM(sec)	Speedup
1500	237.66	26.78	8.87
1600	348.70	31.78	10.97
1700	390.51	35.79	10.91
1800	475.60	41.37	11.50
1900	465.57	45.87	10.15
2000	570.90	50.70	11.26
2000	570.9	50.7	11.26

2100	570	54.91	10.38
2200	678.83	63.59	10.68
2300	710.3	70.36	10.10
2400	766.82	62.5	12.27

Fashion MNIST dataset

Neurons#	Serial pdADMM(sec)	pdADMM(sec)	Speedup
1500	358.68	32.65	10.99
1600	407.71	37.90	10.76
1700	476.79	44.75	10.65
1800	539.51	50.50	10.68
1900	599.42	53.88	11.13
2000	645.87	58.68	11.01
2100	740.39	67.91	10.90
2200	818.58	74.17	11.03

kMNIST dataset

Neurons#	Serial pdADMM(sec)	pdADMM(sec)	Speedup
1500	354.85	32.65	10.87
1600	407.73	37.11	10.99
1700	472.4648	42.58	11.10
1800	539.52	48.78	11.06
1900	596.84	55.56	10.74
2000	660.58	56.10	11.78
2100	737.78	66.95	11.02
2200	806.74	76.16	10.59

CIFAR10 dataset

Neurons#	Serial pdADMM(sec)	pdADMM(sec)	Speedup

1500	326.62	25.00	13.06
1600	374.82	28.96	12.94
1700	433.46	33.99	12.75
1800	485.86	38.66	12.57
1900	544.11	43.10	12.62
2000	572.33	46.90	12.20
2100	602.65	55.25	10.91
2200	732.79	59.27	12.36
2300	784.87	56.26	13.95
2400	854.47	63.1	13.54
CIFAR100 dataset			
Neurons#	Serial pdADMM(sec)	pdADMM(sec)	Speedup
1500	334.55	25.39	13.18
1600	382.24	29.3	13.05
1700	445.23	34	13.09
1800	500.00	38.38	13.03
1900	549.77	43.25	12.71
2000	576.10	42.47	13.56
2100	666.06	47.43	14.04
2200	735.63	52.41	14.04
2300	793.03	56.73	13.98
2400	857.41	62.3	13.76

Table 4.1: The relationship between the speedup of the proposed pdADMM algorithm and the number of neurons on five datasets.

In this experiment, we investigate the speedup of the proposed pdADMM algorithm concerning the number of layers and the number of neurons on large-scale deep

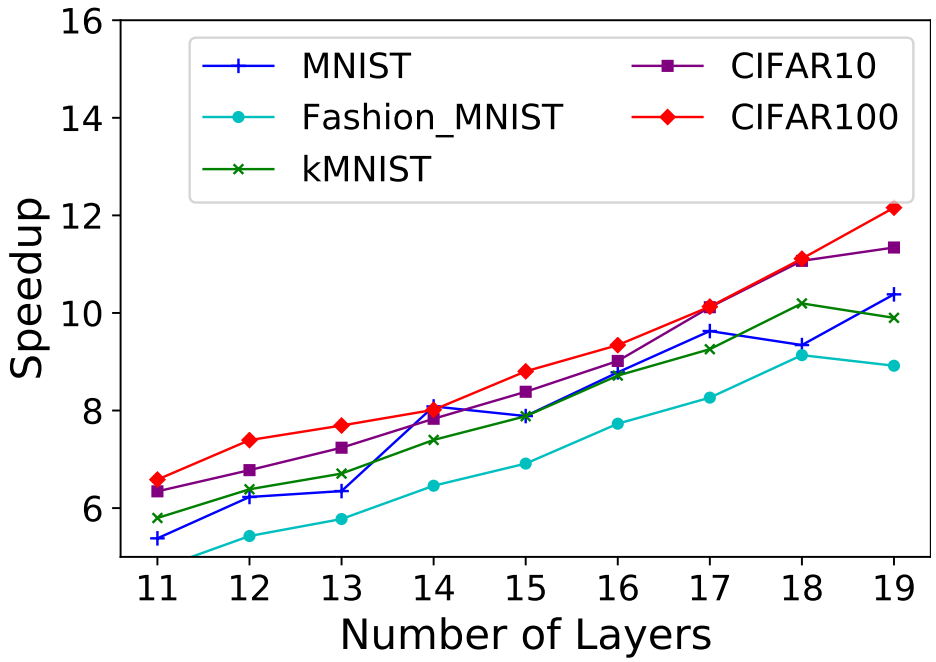
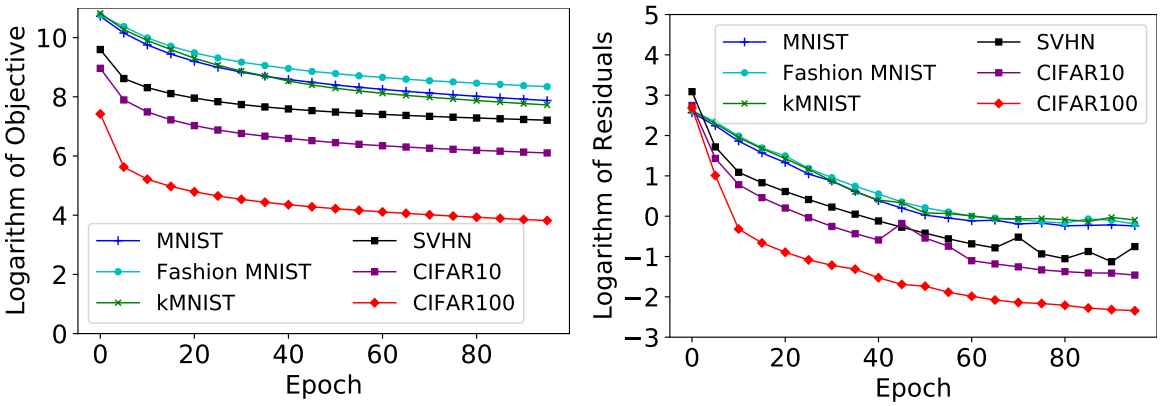


Figure 4.2: The relationship between the speedup of the proposed pdADMM algorithm and the number of layers on five datasets.

neural networks. The activation function was set to ReLU. The loss function was the cross-entropy loss. The running time per epoch was the average of 10 epochs. ρ and ν were both set to 10^{-4}.

Firstly, we investigated the relationship between speedup and the number of layers. We set up an MLP neural network with a different number of hidden layers, which ranges from 11 to 19. The number of neurons in each layer was fixed to 2,400. The SVHN dataset was not tried due to memory issues. Figure 4.2 shows that the speedup increases linearly with the number of layers. Specifically, the speedup reached 11 when 19 hidden layers were trained. This indicates that the deeper a neural network is, the more speedup our proposed pdADMM can gain.

Secondly, the relationship between speedup and the number of neurons was studied. Specifically, we test our proposed pdADMM algorithm on an MLP neural network with 19 hidden layers. The number of neurons in each layer ranges from 1,500 to 2,400. The speedup was shown in Table 4.1 on the MNIST and Fashion MNIST



(a). Objective versus epoch. (b). Residual versus epoch.

Figure 4.3: Convergence curves of the proposed pdADMM algorithm.

datasets. Specifically, the speedup remains stable at around 10 no matter how many neurons were trained. This concludes that the speedup of the proposed pdADMM is independent of the number of neurons.

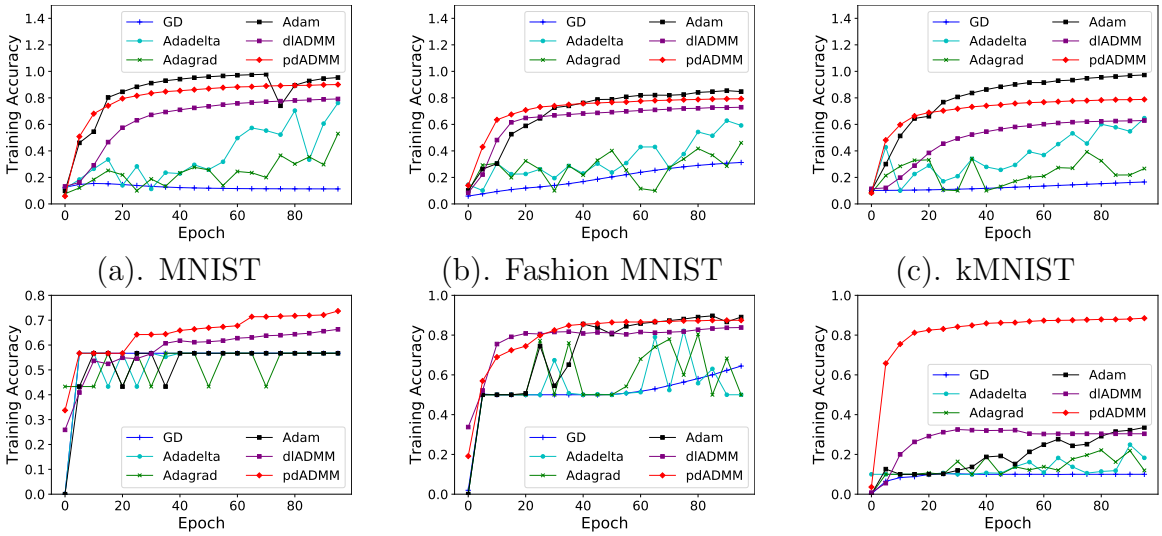
4.5.3 Convergence

To validate the convergence of the proposed pdADMM, we set up an MLP neural network with 9 hidden layers, each of which has 500 neurons. The ReLU was used for the activation function for both network structures. The loss function was set as the cross-entropy loss. The number of the epoch was set to 100. ν and ρ were both set to 0.1. As shown in Figure 4.3, the objective keeps decreasing monotonically on all six datasets, and the residual converges sublinearly to 0, which is consistent with Theorems 7 and 8.

4.5.4 Performance

Experimental Settings

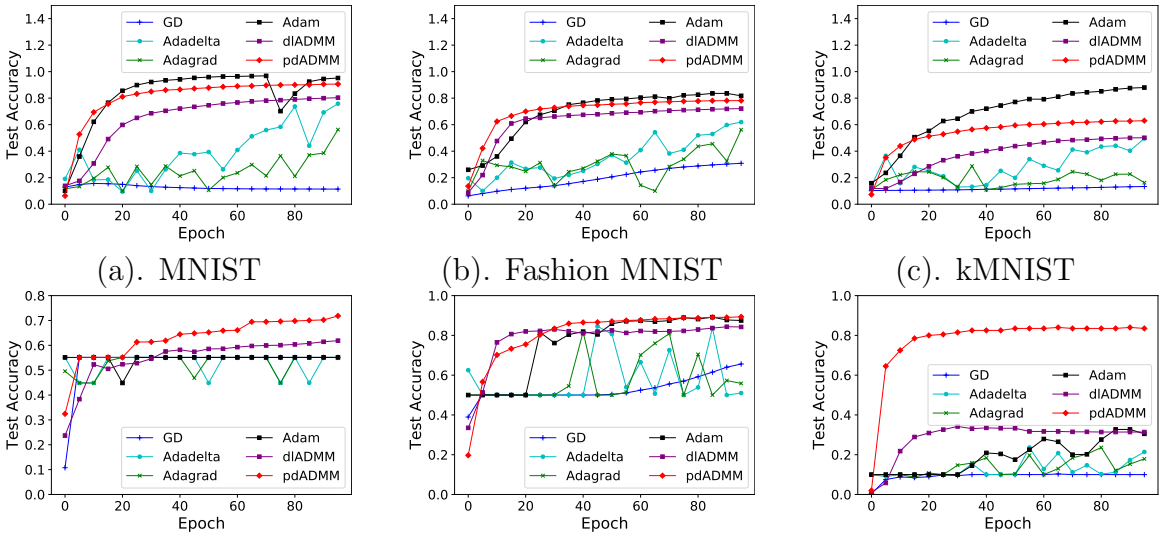
To evaluate the performance of the proposed pdADMM algorithm, we used the same architecture as the previous section. ν and ρ were set to 10^{-4} in order to maximize the performance of training data. The number of the epoch was set to 100. In this



(a). MNIST (b). Fashion MNIST (c). kMNIST

(d). SVHN (e). CIFAR10 (f). CIFAR100

Figure 4.4: Training accuracy of the proposed pdADMM algorithm against all comparison methods on six datasets.



(a). MNIST (b). Fashion MNIST (c). kMNIST

(d). SVHN (e). CIFAR10 (f). CIFAR100

Figure 4.5: Test accuracy of the proposed pdADMM algorithm against all comparison methods on six datasets.

experiment, the full batch was used for training. As suggested by the training strategy in Section 4.3.2, we firstly trained an MLP neural network with five hidden layers, and then all layers were involved in training.

Comparison Methods

GD and its variants are state-of-the-art methods and hence served as comparison methods. For GD-based methods, the full batch dataset is used for training models. All parameters were chosen by maximizing the accuracy of training datasets. The baselines are described as follows:

1. Gradient Descent(GD) [4]. The GD and its variants are the most popular deep learning optimizers, whose convergence has been studied extensively in the literature. The learning rate of GD was set to 0.01.

2. Adaptive learning rate method(Adadelta) [118]. The Adadelta is proposed to overcome the sensitivity to hyperparameter selection. The learning rate of Adadelta was set to 1.

3. Adaptive gradient algorithm(Adagrad) [28]. Adagrad is an improved version of GD: rather than fixing the learning rate during training, it adapts the learning rate to the hyperparameter. The learning rate of Adagrad was set to 0.1.

4. Adaptive momentum estimation(Adam) [52]. Adam is the most popular optimization method for deep learning models. It estimates the first and second momentum to correct the biased gradient and thus makes convergence fast. The learning rate for Adam was set to 0.001.

5. Deep learning Alternating Direction Method of Multipliers(dlADMM) [101]. The dlADMM is an improvement of the previous ADMM implementation [93]. It is guaranteed to converge to a stationary point with a rate of $o(1/k)$. ρ and ν were both set to 10^{-6}.

Performance

In this section, the performance of the proposed pdADMM is analyzed against comparison methods. Figures 4.4 and 4.5 show the training and test accuracy of the proposed pdADMM against comparison methods on six datasets, respectively. The X-axis and Y-axis represent epoch and training accuracy, respectively. Overall, the pdADMM outperformed most of the comparison methods: it performed the best on the SVHN, CIFAR10, and CIFAR100 datasets, while was only secondary to Adam on the MNIST, Fashion MNIST, and kMNIST datasets. The performance gap is particularly obvious in the CIFAR100 dataset. Most GD-type methods suffered from gradient vanishing in the deep neural network, and struggled to find an optimum: for example, GD can only reach 10% training accuracy on the MNIST dataset; Adagrad and Adadelta performed better than GD, but took many epochs to escape saddle points. The dlADMM can avoid the gradient vanishing problem, however, it performed worse than the proposed pdADMM on all datasets. Adam performed the best on three MNIST-like datasets but performed worse than the proposed pdADMM on three other datasets, which are hard to train with high accuracy than three MNIST-like datasets. This indicates that the proposed pdADMM may be more suitable for training hard datasets than Adam.

4.6 Conclusion

The ADMM is considered to be a good alternative to GD for training deep neural networks. In this chapter, we propose a novel parallel deep learning Alternating Direction Method of Multipliers(pdADMM) to achieve layer parallelism. It is guaranteed to converge to a stationary solution under mild conditions. Experiments on benchmark datasets demonstrate that our proposed pdADMM not only leads to a huge speedup for deep MLPs, but also outperforms others on six benchmark datasets.

Chapter 5

The pdADMM-G and pdADMM-G-Q Algorithms

5.1 Introduction

In Chapter 4, we demonstrate the effectiveness of the proposed pdADMM algorithm. In this Chapter, we extend our proposed pdADMM to graph applications, which can be possibly achieved by Graph Neural Networks(GNNs). They have accomplished state-of-the-art performance in various graph applications such as node classification and link prediction. This is because they handle graph-structured data via aggregating neighbor information and extending operations and definitions of the deep learning approach [53]. However, their performance has significantly been restricted via their depths due to the over-smoothing problem(i.e. the representations of different nodes in a graph tend to be similar when stacking multiple layers) [15], and the over-squashing problem(i.e. the information flow among distant nodes distorts along the long-distance interactions) [96]. These challenges still exist even though some models such as GraphSAGE [35] have been proposed to alleviate them.

On the other hand, the Graph Augmented Multi-Layer Perceptron(GA-MLP)

models have recently received fast-increasing attention as an alternative to deal with the aforementioned drawbacks of conventional GNNs via the augmentation of graph features. GA-MLP models augment node representations of graphs and feed them into Multi-Layer Perceptron(MLP) models. Compared with GNNs, GA-MLP models are more resistant to the over-smoothing problem [15] and therefore demonstrate outstanding performance. For example, Wu et al. showed that a two-layer GA-MLP approximates the performance of the GNN models on multiple datasets [110].

GA-MLP models are supposed to perform better with the increase of their depths. However, similar to GNNs, GA-MLP models still suffer from the gradient vanishing problem, which is caused by the mechanism of the classic backpropagation algorithm. This is because gradient signals diminish during the transmission among deep layers. Moreover, while the models go deeper, efficiency will become an issue, especially for medium- and large-size graphs. Compared to the data such as images and texts, where identically and independently distributed(i.i.d.) samples are assumed, efficiency issues in graph data are much more difficult to handle due to the dependency among data samples(i.e., nodes in graphs). Such dependency seriously troubles the effectiveness of using typical acceleration techniques such as sampling-based methods, and data-parallelism distributed learning in solving the efficiency issue. Therefore, parallelizing the computation along layers is a natural workaround, but the backpropagation prevents the gradients of different layers from being calculated in parallel. This is because the calculation of the gradient in one layer is dependent on its previous layers.

To handle these challenges, we extend the proposed pdADMM algorithm to train large-scale GA-MLP models, named the graph pdADMM(pdADMM-G) algorithm, and the quantized graph pdADMM(pdADMM-G-Q) algorithm reduces communication costs of the pdADMM-G algorithm by the quantization technique. Our contributions to this chapter include:

- **We propose a novel reformulation of GA-MLP models**. It splits a neural

network into independent layer partitions and allow for ADMM to achieve model parallelism.

- **We propose a novel pdADMM-G framework to train a GA-MLP model**. All subproblems generated by the ADMM algorithm are discussed. The extended pdADMM-G-Q algorithm reduces communication costs by introducing the quantization technique.

- **We provide the theoretical convergence guarantee of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm**. Specifically, they converge to a(quantized) stationary point of GA-MLP models when the hyperparameters are sufficiently large, and their sublinear convergence rates are $o(1/k)$.

- **We conduct extensive experiments to show the effectiveness of two proposed algorithms**. Experiments on nine benchmark datasets show the convergence, the massive speedup of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm, as well as their outstanding performance when compared with all state-of-the-art optimizers. Moreover, the pdADMM-G-Q algorithm reduces communication overheads by up to 45%.

The organization of this chapter is shown as follows: In Section 5.2, we summarize recent related research work to this chapter. In Section 5.3, we propose the pdADMM-G algorithm and the pdADMM-G-Q algorithm to train deep GA-MLP models. Section 5.4 details the convergence properties of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm. Extensive experiments on nine benchmark datasets to demonstrate the convergence, speedup, communication savings, and outstanding performance of the pdADMM-G algorithm and the pdADMM-G-Q algorithm are shown in Section 5.5, and Section 5.6 concludes this work.

5.2 Related Work

This section summarizes existing literature related to this chapter. One branch of related research is distributed deep learning, which has been discussed in Section 4.2 in Chapter 4. The other branch of related papers are deep learning on graph, which is shown as follows:

Deep Learning on Graphs: Graphs are ubiquitous structures and are popular in real-world applications. There is a surge of interest to apply deep learning techniques to graphs. For a comprehensive summary please refer to [111]. It classified existing GNN models into four categories: Recurrent Graph Neural Networks(RecGNNs), Convolutional Graph Neural Networks(ConvGNNs), Graph Autoencoders(GAEs), and Spatial-Temporal Graph Neural Networks(STGNNs).
RecGNNs learn node representation with recurrent neural networks via the message passing mechanisms [23, 29, 63]; ConvGNNs generalize the operations of convolution to graph data and stack multiple convolution layers to extract high-level node features [7, 25, 38]; GAEs encode node information into a latent space and reconstruct graphs from the encoded node representation [8, 78, 97]; the idea of STGNNs is to capture spatial dependency and temporal dependency simultaneously [48, 62, 87].

5.3 The pdADMM-G Algorithm

We propose the pdADMM-G algorithm to solve GA-MLP models in this section. Specifically, Section 5.3.1 formulates the GA-MLP model training problem, and Section 5.3.2 proposes the pdADMM-G algorithm. Section 5.3.3 extends the proposed pdADMM-G algorithm to the pdADMM-G-Q algorithm for quantization.

Notations	Descriptions
L	Number of layers.
W_l	The weight matrix for the l-th layer.
b_l	The intercept vector for the l-th layer.
z_l	The auxiliary variable of the linear mapping for the l-th layer.
$f_l(z_l)$	The nonlinear activation function for the l-th layer.
p_l	The input for the l-th layer.
q_l	The output for the l-th layer.
X	The node representation of the graph.
A	The adjacency matrix of the graph.
y	The predefined label vector.
$R(z_L, y)$	The risk function for the L-th layer.
n_l	The number of neurons for the l-th layer.
u_l	The dual variable for the l-th layer.

Table 5.1: Important Notations

5.3.1 Problem Formulation

Consider a graph $G = (V, E)$, where V and E are sets of nodes and edges, respectively, $|V|$ is the number of nodes, let $\Psi = \{\psi_1(A), \cdots, \psi_K(A)\}$ be a set of(usually multi-hop) operators $\psi_i(A) : \mathbb{R}^{|V|} \to \mathbb{R}^{|V|}(i = 1, \cdots, K)$ that are functions of the adjacency matrix $A \in \{0, 1\}^{|V|\times|V|}$, and $\mathbb{R}^{|V|}$ is the domain of $\psi_i(A)$ $(i = 1, \cdots, K)$. $x_k = H\psi_k(A)$ is the augmentation of node features by the k-hop operator, where $H \in \mathbb{R}^{d\times|V|}$ is a matrix of node features, and d is the dimension of features. $x_k(k = 1, \cdots, K)$ are stacked into $x = [x_1; \cdots; x_K]$ by column. Then the GA-MLP training problem is formulated as follows [15]:

Problem 5.

$$\min_{W_l, z_l, p_l} R(z_L, y), \quad s.t.\ z_l = W_l p_l,\ p_{l+1} = f_l(z_l)(l = 1, \cdots, L-1),$$

where $p_1 = X \in \mathbb{R}^{n_0\times|V|}$ is the input of deep GA-MLP models, where $n_0 = Kd$ is the dimension of input and y is a predefined label vector. $p_l \in \mathbb{R}^{n_l\times|V|}$ is the input of the l-th layer, also the output for the $(l-1)$-th layer, and n_l is the number of neurons for the l-th layer. $R(z_L, y)$ is a risk function of the L-th layer, which is convex and

continuous; $z_l = W_l p_l$ and $p_{l+1} = f_l(z_l)$ are linear and nonlinear mappings of the l-th layer, respectively, and $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix of the l-th layer.

In Problem 5, Ψ can be considered as a prepossessing step to augment node features via A, and hence it is predefined. One common choice can be $\Psi = \{I, A, A^2, \cdots, A^{K-1}\}$.

We apply the pdADMM algorithm in Chapter 4 to Problem 5 to realize model parallelism training. To achieve this, we relax Problem 5 to Problem 6 as follows:

Problem 6.

$$\min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}} F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}) = R(z_L, y) + (\nu/2)\Big(\sum_{l=1}^{L} \|z_l - W_l p_l\|_2^2 + \sum_{l=1}^{L-1} \|q_l - f_l(z_l)\|_2^2\Big),$$
$$s.t.\ p_{l+1} = q_l.$$

where $\mathbf{p} = \{p_l\}_{l=1}^{L}$, $\mathbf{W} = \{W_l\}_{l=1}^{L}$, $\mathbf{z} = \{z_l\}_{l=1}^{L}$, $\mathbf{q} = \{q_l\}_{l=1}^{L-1}$, and $\nu > 0$ is a tuning parameter. We reduce layer dependency by splitting the output of the l-th layer and the input of the$(l+1)$-th layer into two variables p_{l+1} and q_l, respectively. As $\nu \to \infty$, Problem 6 approaches Problem 5.

5.3.2 The pdADMM-G Algorithm

The high-level overview of the pdADMM-G algorithm is shown in Figure 5.1. Specifically, the inputs of GA-MLP models are augmented by $H\psi_k(A)$ $(k = 1, \cdots, K)$, and then GA-MLP models are split into multiple layers, each of which can be optimized by an independent client. Therefore, layerwise training can be implemented in parallel. Moreover, the gradient vanishing problem can be avoided in this way. This is because the accumulated gradient calculated by the backpropagation algorithm is split into layerwise components.

Now we follow the ADMM routine to solve Problem 6. The augmented Lagrangian

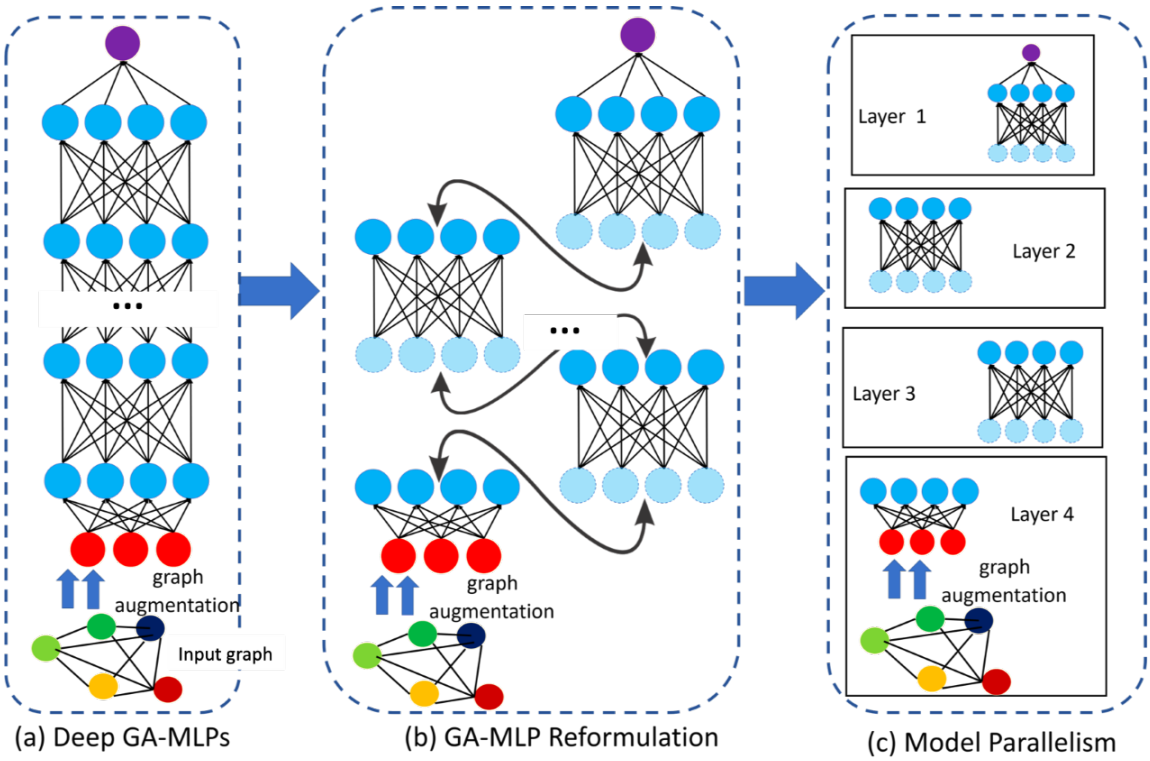


Figure 5.1: The overview of the proposed pdADMM-G optimization algorithm.

function is formulated mathematically as follows:

$$
\begin{aligned}
&L_\rho(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q},\mathbf{u})\\
&=F(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q})+\sum_{l=1}^{L-1}(u_l^T(p_{l+1}-q_l)+(\rho/2)\|p_{l+1}-q_l\|_2^2)\\
&=R(z_L,y)+\phi(p_1,W_1,z_1)+\sum_{l=2}^{L}\phi(p_l,W_l,z_l,q_{l-1},u_{l-1})+(\nu/2)\sum_{l=1}^{L-1}\|q_l-f_l(z_l)\|_2^2,
\end{aligned}
$$

where $\phi(p_1,W_1,z_1)=(\nu/2)\|z_1-W_1p_1\|_2^2$, $\phi(p_l,W_l,z_l,q_{l-1},u_{l-1})=(\nu/2)\|z_l-W_lp_l\|_2^2+u_{l-1}^T(p_l-q_{l-1})+(\rho/2)\|p_l-q_{l-1}\|_2^2$, $u_l(l=1,\cdots,L-1)$ are dual variables, $\rho>0$ is a parameter, and $\mathbf{u}=\{u_l\}_{l=1}^{L-1}$. The detail of the pdADMM-G algorithm is shown in Algorithm 6. Specifically, Lines 5-8 update primal variables $\mathbf{p}$, $\mathbf{W}$, $\mathbf{z}$ and $\mathbf{q}$, respectively, while Line 10 updates the dual variable $\mathbf{u}$. The details of all subproblems are shown in Section 4.3 in Chapter 4.

Our proposed pdADMM-G algorithm can be efficient for training deep GA-MLP

Algorithm 6: The proposed pdADMM-G algorithm

Require: y, $p_1 = X$, ρ, ν.
Ensure: $\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}$.
Initialize $k = 0$.
while $\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k$ not converged **do**
$p_l^{k+1} \leftarrow \arg\min_{p_l} L_\rho(\mathbf{p}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ for different l in parallel.
$W_l^{k+1} \leftarrow \arg\min_{W_l} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ for different l in parallel.
$z_l^{k+1} \leftarrow \arg\min_{z_l} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}, \mathbf{q}^k, \mathbf{u}^k)$ for different l in parallel.
$q_l^{k+1} \leftarrow \arg\min_{q_l} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}, \mathbf{u}^k)$ for different l in parallel.
$r_l^k \leftarrow p_{l+1}^{k+1} - q_l^{k+1} (l = 1, \cdots, L)$ in parallel # Compute residuals.
$u_l^{k+1} \leftarrow u_l^k + \rho(p_{l+1}^{k+1} - q_l^{k+1})$ for different l in parallel.
$k \leftarrow k + 1$.
end while
Output $\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}$.

models via the greedy layerwise training strategy [3]. Specifically, we begin by training a shallow GA-MLP model. Next, more layers are increased to the GA-MLP model and their parameters are trained, then we introduce even more layers and iterate this process until the whole deep GA-MLP model is included. The pdADMM-G algorithm can achieve excellent performance as well as reduce training costs by this strategy.

Last but not least, we compare the computational costs of the proposed pdADMM-G algorithm with the state-of-the-art backpropagation algorithm, on which the gradient descent is based. We show that they share the same level of computational costs. For the backpropagation algorithm, the most costly operation is the matrix multiplication $z_l = W_l p_l$ in the forward pass, where $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$ and $p_l \in \mathbb{R}^{n_{l-1} \times |V|}$, which requires a time complexity of $O(n_l n_{l-1}|V|)$ [21]; for the proposed pdADMM-G algorithm, the most costly operation is to compute the derivative $\nabla_{W_l}\phi$, and it also involves the matrix multiplication, and hence its time complexity is again $O(n_l n_{l-1}|V|)$. However, the proposed pdADMM-G algorithm trains the whole GA-MLP model in a model parallelism fashion [103], and therefore all computational costs can be split into different independent clients for parallel training; whereas the backpropagation algorithm is implemented sequentially, and thus it is less efficient than the proposed

pdADMM-G algorithm.

5.3.3 Quantization Extension of pdADMM-G(pdADMM-G-Q)

In the proposed pdADMM-G algorithm, p_l and q_l are transmitted back and forth among layers(i.e. clients). However, the communication overheads of p_l and q_l surge for a large-scale graph G with millions of nodes. To alleviate this challenge, the quantization technique is commonly utilized to reduce communication costs by mapping continuous values into a discrete set [42]. In other words, p_l is required to fit into a countable set Δ, which is shown as follows:

Problem 7.

$$\begin{aligned}\min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}} F(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}) = R(z_L,y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l - W_l p_l\|_2^2 \\ + \sum\nolimits_{l=1}^{L-1} \|q_l - f_l(z_l)\|_2^2), \\ s.t.\ p_{l+1} = q_l,\ p_l \in \Delta = \{\delta_1, \cdots, \delta_m\},\end{aligned}$$

where $\delta_i (i = 1, \cdots, m) \in \Delta$ are quantized values, which can be integers or low-precision values. $m = |\Delta|$ is the cardinality of Δ. To address Problem 7, we rewrite it into the following form:

$$\begin{aligned}\min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}} R(z_L,y) + \sum\nolimits_{l=2}^{L} \mathbb{I}(p_l) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l - W_l p_l\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l - f_l(z_l)\|_2^2), \\ s.t.\ p_{l+1} = q_l,\end{aligned}$$

where the indicator function $\mathbb{I}(p_l)$ is defined as follows: $\mathbb{I}(p_l) = 0$ if $p_l \in \Delta$, and $\mathbb{I}(p_l) = +\infty$ if $p_l \notin \Delta$. The augmented Lagrangian of Problem 7 is shown as follows:

$$\beta_\rho(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q},\mathbf{u}) = L_\rho(\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q},\mathbf{u}) + \sum\nolimits_{l=2}^{L} \mathbb{I}(p_l),$$

where L_ρ is the augmented Lagrangian of Problem 6. The extended pdADMM-G-Q algorithm follows the same routine as the pdADMM-G algorithm, where L_ρ is replaced with β_ρ. Obviously, the only difference between the pdADMM-G-Q algorithm and the pdADMM-G algorithm is the p_l-subproblem, which is outlined in the following:

$$p_l^{k+1} \leftarrow \arg\min_{p_l} U_l(p_l; \tau_l^{k+1}) + \mathbb{I}(p_l), \tag{5.1}$$

where U_l follows Equation (4.1). The solution to Equation (5.1) is [42]: $p_l^{k+1} \leftarrow \arg\min_{\delta\in\Delta} \|\delta - (p_l^k - \nabla_{p_l^k}\phi(p_l^k, W_l^k, b_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)/\tau_l^{k+1})\|$.

For the pdADMM-G-Q algorithm, the variable $\mathbf{p}$ is only required to be quantized(i.e. $p_l \in \Delta$) when the p_l-subproblem is solved(i.e. Equation (5.1)), and the variable $\mathbf{q}$ can be any real number when it is updated. However, $\mathbf{q}$ is guaranteed to fit into Δ by the linear constraint $p_{l+1} = q_l$. This design is convenient for the convergence analysis, which is detailed in the next section. One variant of the pdADMM-G-Q algorithm is to quantize $\mathbf{p}$ and $\mathbf{q}$(i.e. $p_l, q_l \in \Delta$) when they are updated. In this case, the solution to q_l subproblem is $q_l^{k+1} \leftarrow \arg\min_{\delta\in\Delta} \|\delta - (\rho p_{l+1}^{k+1} + u_l^k + \nu f_l(z_l^{k+1}))/(\rho+\nu)\|$.

5.4 Convergence Analysis

In this section, the theoretical convergence of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm is provided. Due to space limit, we only provide sketches of proofs in this section, and their details are available in Section D.1 in the Appendix. Our problem formulations are more difficult than existing ADMM literature: the term $\|q_l - f_l(z_l)\|_2^2$ is coupled in the objective, while it is separable in the existing ADMM formulations. To address this, we impose a mild condition that

$\partial f_l(z_l)$ is bounded in Assumption 6, and prove that u_l is controlled via q_l and z_l in Lemma 25 in Section D.1 in the Appendix.

Firstly, the proper function, Lipschitz continuity, and coercivity are defined as follows:

Definition 3 (Proper Functions)**.** [84]. For a convex function $g(x) : \mathbb{R} \to \mathbb{R} \bigcup \{\pm\infty\}$, g is called proper if $\forall x \in \mathbb{R}$, $g(x) > -\infty$, and $\exists x_0 \in \mathbb{R}$ such that $g(x_0) < +\infty$.

Definition 4. (Lipschitz Continuity) A function $g(x)$ is Lipschitz continuous if there exists a constant $D > 0$ such that $\forall x_1, x_2$, the following holds

$$\|g(x_1) - g(x_2)\| \leq D\|x_1 - x_2\|.$$

Definition 5. (Coercivity) A function $h(x)$ is coerce over the feasible set $\mathscr{F}$ means that $h(x) \to \infty$ if $x \in \mathscr{F}$ and $\|x\| \to \infty$.

Next, the definition of a quantized stationary point [42] is shown as follows:

Definition 6. (Quantized Stationary Point) The p_l is a quantized stationary point of of Problem 7 if there exists $\tau > 0$ such that

$$p_l \in \arg\min_{\delta \in \Delta} \|\delta - (p_l - \nabla_{p_l} F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q})/\tau)\|.$$

The quantized stationary point is an extension of the stationary point in the discrete setting, and any global solution p_l to Problem 7 is a quantized stationary point to Problem 7(Lemma 3.7 in [42]). Then the following assumption is required for convergence analysis.

Assumption 6. $f_l(z_l)$ is Lipschitz continuous with coefficient $S > 0$, $R(z_L, y)$ is proper, and $F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q})$ is coercive. Moreover, $\partial f_l(z_l)$ is bounded, i.e. there exists $M > 0$ such that $\|\partial f_l(z_l)\| \leq M$.

Assumption 6 is mild to satisfy: most common activation functions such as Rectified Linear Unit(ReLU) [103] and leaky ReLU[114] satisfy Assumption 6. The risk function $R(z_L, y)$ is only required to be proper, which shows that the convergence condition of our proposed pdADMM-G is milder than that of the dlADMM, which requires $R(z_L, y)$ to be Lipschitz differentiable [101]. Due to the space limit, detailed proofs are provided in Section D.1 in the Appendix. The technical proofs follow a similar routine as dlADMM [101]. The difference consists in the fact that the dual variable u_l is controlled by q_l and z_l(Lemma 26 in Section D.1 in the Appendix), which holds under Assumption 6, while u_l can be controlled only by z_l in the convergence proof of dlADMM. The first lemma shows that the objective keeps decreasing when ρ is sufficiently large.

Lemma 8 (Objective Decrease)**.** For both the pdADMM-G algorithm and the pdADMM-G-Q algorithm, if $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, there exist $C_1 = \nu/2 - 2\nu^2 S^2/\rho > 0$ and $C_2 = \rho/2 - 2\nu^2/\rho - \nu/2 > 0$ such that it holds for any $k \in \mathbb{N}$ that

$$
\begin{aligned}
&L_\rho(\mathbf{p}^k,\mathbf{W}^k,\mathbf{z}^k,\mathbf{q}^k,\mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1},\mathbf{W}^{k+1},\mathbf{z}^{k+1},\mathbf{q}^{k+1},\mathbf{u}^{k+1}) \\
&\geq \sum_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1}-p_l^k\|_2^2 + \sum_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1}-W_l^k\|_2^2 + \sum_{l=1}^{L-1} C_1\|z_l^{k+1}-z_l^k\|_2^2 \\
&+(\nu/2)\|z_L^{k+1}-z_L^k\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{k+1}-q_l^k\|_2^2, \qquad (5.2) \\
&\beta_\rho(\mathbf{p}^k,\mathbf{W}^k,\mathbf{z}^k,\mathbf{q}^k,\mathbf{u}^k) - \beta_\rho(\mathbf{p}^{k+1},\mathbf{W}^{k+1},\mathbf{z}^{k+1},\mathbf{q}^{k+1},\mathbf{u}^{k+1}) \\
&\geq \sum_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1}-W_l^k\|_2^2 + \sum_{l=1}^{L-1} C_1\|z_l^{k+1}-z_l^k\|_2^2 + (\nu/2)\|z_L^{k+1}-z_L^k\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_2\|q_l^{k+1}-q_l^k\|_2^2. \qquad (5.3)
\end{aligned}
$$

Sketch of Proof: They can be proven via the optimality conditions of all subproblems, and Assumption 6. □

Lemma 9 shows that the objective is bounded from below when ρ is large enough, and all variables are bounded.

Lemma 9 (Bounded Objective)**.** (1). For the pdADMM-G algorithm, if $\rho > \nu$, then $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is lower bounded. Moreover, $\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k$,and $\mathbf{u}^k$ are bounded, i.e. there exist $\mathbb{N}_\mathbf{p}$, $\mathbb{N}_\mathbf{W}$, $\mathbb{N}_\mathbf{z}$, $\mathbb{N}_\mathbf{q}$, and $\mathbb{N}_\mathbf{u} > 0$, such that $\|\mathbf{p}^k\| \leq \mathbb{N}_\mathbf{p}$, $\|\mathbf{W}^k\| \leq \mathbb{N}_\mathbf{W}$, $\|\mathbf{z}^k\| \leq \mathbb{N}_\mathbf{z}$, $\|\mathbf{q}^k\| \leq \mathbb{N}_\mathbf{q}$, and $\|\mathbf{u}^k\| \leq \mathbb{N}_\mathbf{u}$.
(2). For the pdADMM-G-Q algorithm, if $\rho > \nu$, then $\beta_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is lower bounded. Moreover, $\mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k$,and $\mathbf{u}^k$ are bounded, i.e. there exist $\overline{\mathbb{N}}_\mathbf{W}$, $\overline{\mathbb{N}}_\mathbf{z}$, $\overline{\mathbb{N}}_\mathbf{q}$, and $\overline{\mathbb{N}}_\mathbf{u} > 0$, such that $\|\mathbf{W}^k\| \leq \overline{\mathbb{N}}_\mathbf{W}$, $\|\mathbf{z}^k\| \leq \overline{\mathbb{N}}_\mathbf{z}$, $\|\mathbf{q}^k\| \leq \overline{\mathbb{N}}_\mathbf{q}$, and $\|\mathbf{u}^k\| \leq \overline{\mathbb{N}}_\mathbf{u}$.

Sketch of Proof: We only show the sketch proof of(1) because(2) follows the same routine as(1). In order to prove the boundness of L_ρ, we should prove the following:

$$\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&\geq F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'}) + ((\rho - \nu)/2)\|p_{l+1}^k - q_l^k\|_2^2 \\
&> -\infty,
\end{aligned}$$

where $p_{l+1}^k = q_l^{'}$. Therefore, $F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'})$ and $((\rho - \nu)/2)\|p_{l+1}^k - q_l^k\|_2^2$ are upper bounded by $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ and hence $L_\rho(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$(Lemma 8). The boundness of variables can be obtained via Assumption 6. □

Based on Lemmas 8 and 9, the following theorem ensures that the objective is convergent.

Theorem 9 (Convergent Objective)**.** (1). For the pdADMM-G algorithm, if $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, then $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent. Moreover, $\lim_{k\to\infty} \|\mathbf{p}^{k+1} - \mathbf{p}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{q}^{k+1} - \mathbf{q}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0$.
(2). For the pdADMM-G-Q algorithm, if $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, then $\beta_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent. Moreover, $\lim_{k\to\infty} \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{q}^{k+1} - \mathbf{q}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0$.

Sketch of Proof: This theorem can be derived by taking the limit on both sides of Inequality (5.2). □

The third lemma guarantees that the subgradient of the objective is upper bounded, which is stated as follows:

Lemma 10 (Bounded Subgradient)**.** (1). For the pdADMM-G algorithm, there exists a constant $C > 0$ and $g^{k+1} \in \partial L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}\|g^{k+1}\| \leq C(&\|\mathbf{p}^{k+1} - \mathbf{p}^k\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| \\ &+ \|\mathbf{u}^{k+1} - \mathbf{u}^k\|).\end{aligned}$$

(2). For the pdADMM-G-Q algorithm, there exists a constant $\overline{C} > 0$, $\overline{g}_{\mathbf{W}}^{k+1} \in \nabla_{\mathbf{W}^{k+1}} \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$,
$\overline{g}_{\mathbf{z}}^{k+1} \in \partial_{\mathbf{z}^{k+1}} \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$,
$\overline{g}_{\mathbf{q}}^{k+1} \in \nabla_{\mathbf{q}^{k+1}} \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$,
$\overline{g}_{\mathbf{u}}^{k+1} \in \nabla_{\mathbf{u}^{k+1}} \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$ such that

$$\begin{aligned}\|\overline{g}_{\mathbf{W}}^{k+1}\| &\leq \overline{C}(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| + \|\mathbf{u}^{k+1} - \mathbf{u}^k\|), \\ \|\overline{g}_{\mathbf{z}}^{k+1}\| &\leq \overline{C}(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| + \|\mathbf{u}^{k+1} - \mathbf{u}^k\|), \\ \|\overline{g}_{\mathbf{q}}^{k+1}\| &\leq \overline{C}(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| + \|\mathbf{u}^{k+1} - \mathbf{u}^k\|), \\ \|\overline{g}_{\mathbf{u}}^{k+1}\| &\leq \overline{C}(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{q}^{k+1} - \mathbf{q}^k\| + \|\mathbf{u}^{k+1} - \mathbf{u}^k\|).\end{aligned}$$

Sketch of Proof: To prove this lemma, the subgradient is proven to be upper bounded by the linear combination of $\|\mathbf{p}^{k+1} - \mathbf{p}^k\|$, $\|\mathbf{W}^{k+1} - \mathbf{W}^k\|$, $\|\mathbf{z}^{k+1} - \mathbf{z}^k\|$, $\|\mathbf{q}^{k+1} - \mathbf{q}^k\|$, and $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|$. □

Now based on Theorem 9, and Lemma 10, the convergence of the pdADMM-G algorithm to a stationary point is presented in the following theorem.

Theorem 10 (Convergence of the pdADMM-G algorithm)**.** If $\rho > \max(4\nu S^2, (\sqrt{17}+$

$1)\nu/2)$, then for the variables $(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}, \mathbf{u})$ in Problem 6, starting from any $(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$, and any limit point is a stationary point of Problem 6. That is,
$0 \in \partial L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$. In other words,

$$p_{l+1}^* = q_l^*,\ \nabla_{\mathbf{p}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0,\ \nabla_{\mathbf{W}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0,$$
$$0 \in \partial_{\mathbf{z}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*), \nabla_{\mathbf{q}^*} L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0.$$

Sketch of Proof: This theorem can be derived directly from Lemma 9 and Lemma 10. □

Theorem 10 shows that our proposed pdADMM-G algorithm converges for sufficiently large ρ, which is consistent with previous literature [101]. Similarly, the convergence of the proposed pdADMM-G-Q algorithm is shown as follows:

Theorem 11 (Convergence of the pdADMM-G-Q algorithm)**.** If $\rho > \max(4\nu S^2, (\sqrt{17}+1)\nu/2)$, then for the variables $(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}, \mathbf{u})$ in Problem 7, starting from any $(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$, and any limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ is a stationary point of Problem 7. Moreover, if τ_l^{k+1} is bounded, then $\mathbf{p}^*$ is a quantized stationary point of Problem 7. That is

$$p_{l+1}^* = q_l^*,\ \nabla_{\mathbf{W}^*} \beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0, 0 \in \partial_{\mathbf{z}^*} \beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*),$$
$$\nabla_{\mathbf{q}^*} \beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0, p_l^* \in \arg\min_{\delta \in \Delta} \|\delta - (p_l^* - \nabla_{p_l^*} F(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*)/\tau_l^*)\|.$$

where τ_l^* is a limit point of τ_l^k.

Sketch of Proof: This theorem is proven using a similar procedure as Theorem 10, and the definition of the quantized stationary point. □

The only difference between Theorems 10 and 11 is that $\mathbf{p}^*$ is a stationary point in Problem 6 and a quantized stationary point in Problem 7. Next, the following

theorem ensures the sublinear convergence rate $o(1/k)$ of the proposed pdADMM-G algorithm and the pdADMM-G-Q algorithm.

Theorem 12 (Convergence Rate)**.** (1). For the pdADMM-G algorithm and a sequence $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$, define $c_k = \min_{0\leq i\leq k}(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1}-p_l^i\|_2^2$ $+\sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2+\sum_{l=1}^{L-1}C_1\|z_l^{i+1}-z_l^i\|_2^2+(\nu/2)\|z_L^{i+1}-z_L^i\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{i+1}-q_l^i\|_2^2)$ where $C_1=\nu/2-2\nu^2S^2/\rho>0$ and $C_2=\rho/2-2\nu^2/\rho-\nu/2>0$, then the convergence rate of c_k is $o(1/k)$.

(2). For the pdADMM-G-Q algorithm and a sequence $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$, define $d_k = \min_{0\leq i\leq k}(\sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2+\sum_{l=1}^{L-1}C_1\|z_l^{i+1}-z_l^i\|_2^2+(\nu/2)\|z_L^{i+1}-z_L^i\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{i+1}-q_l^i\|_2^2)$ where $C_1=\nu/2-2\nu^2S^2/\rho>0$ and $C_2=\rho/2-2\nu^2/\rho-\nu/2>0$, then the convergence rate of d_k is $o(1/k)$.

Sketch of Proof: (1). In order to prove the convergence rate $o(1/k)$, c_k satisfies three conditions:(a) $c_k \geq c_{k+1}$, (b) $\sum_{k=0}^{\infty} c_k$ is bounded, and(c) $c_k \geq 0$.

(2). d_k can be proven using a similar procedure as(1). □

5.5 Experiments

In this section, we evaluate the performance of the proposed pdADMM-G algorithm and the proposed pdADMM-G-Q algorithm on GA-MLP models using nine benchmark datasets. Convergence and computational overheads are demonstrated on different datasets. Speedup and test performance are compared with several state-of-the-art optimizers. All experiments were conducted on the Amazon Web Services(AWS) p2.16xlarge instance, with 16 NVIDIA K80 GPUs, 64vCPUs, a processor Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, and 732GiB of RAM.

Dataset	Node#	Edge#	Class#	Feature#	Training Set#	Validation Set#	Test Set#
Cora	2485	10 556	7	1433	140	500	1000
PubMed	19 717	88 648	3	500	60	500	1000
Citeseer	2110	9104	6	3703	120	500	1000
Amazon Computers	13 381	491 722	10	767	200	1000	1000
Amazon Photo	7487	238 162	8	745	160	1000	1000
Coauthor CS	18 333	163 788	15	6805	300	1000	1000
Coauthor Physics	34 493	495 924	5	8415	100	1000	1000
Flickr	89 250	899 756	7	500	44 625	22 312	22 313
Ogbn-Arxiv	169 343	1 166 243	40	128	90 941	29 799	48 603

Table 5.2: Statistics of eight benchmark datasets.

5.5.1 Datasets and Settings

Nine benchmark datasets were used for experimental evaluation, whose statistics are shown in Table 5.2. Each dataset is split into a training set, a validation set, and a test set. The details are below:

1. Cora [86]. The Cora dataset consists of 2708 scientific publications classified into one of seven classes. The citation network consists of 5429 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words.

2. PubMed [86]. PubMed comprises 30M+ citations for biomedical literature that have been collected from sources such as MEDLINE, life science journals, and published online e-books. It also includes links to text content from PubMed Central and other publishers' websites.

3. Citeseer [86]. The Citeseer dataset was collected from the Tagged.com social network website. It contains 5.6 million users and 858 million links between them. Each user has 4 features and is manually labeled as "spammer" or "not spammer". Each

link represents an action between two users and includes a timestamp and a type. The network contains 7 anonymized types of links. The original task on the dataset is to identify(i.e., classify) the spammer users based on their relational and non-relational features.

4. Amazon Computers and Amazon Photo [68]. Amazon Computers and Amazon Photo are segments of the Amazon co-purchase graph, where nodes represent goods, edges indicate that two goods are frequently bought together, node features are bag-of-words encoded product reviews, and class labels are given by the product category.

5. Coauthor CS and Coauthor Physics [89]. Coauthor CS and Coauthor Physics are co-authorship graphs based on the Microsoft Academic Graph from the KDD Cup 2016 challenge 3. Here, nodes are authors, that are connected by an edge if they co-authored a paper; node features represent paper keywords for each author's papers, and class labels indicate the most active fields of study for each author.

6. Flickr [119]. In Flickr, one node in the graph represents one image uploaded to Flickr. If two images share some common properties(e.g., same geographic location, same gallery, comments by the same user, etc.), there is an edge between the nodes of these two images. Node features are bag-of-word representation of the images and labels are classes of images.

7. Ogbn-Arxiv [41]. The Ogbn-Arxiv dataset is a directed graph, representing the citation network between all Computer Science(CS) ARXIV papers indexed by MAG. Each node is an ARXIV paper and each directed edge indicates that one paper cites another one. Each paper comes with a 128-dimensional feature vector obtained by averaging the embeddings of words in its title and abstract. In addition, all papers are also associated with the year that the the corresponding paper was published.

When it comes to experimental settings, we set $K = 4$ for the multi-hop operator Ψ, and defined a diagonal degree matrix D where $D_{ii} = \sum_{j=1}^{|V|} A_{ij}$, and a renormalized adjacency matrix $\tilde{A} = (D + I)^{-1/2}(A + I)(D + I)^{-1/2} \in \mathbb{R}^{|V|\times|V|}$[53]. Moreover, we

set $\Psi = \{I, \tilde{A}, \tilde{A}^2, \tilde{A}^3\}$ [15]. For all GA-MLP models, the activation function was set to ReLU. The loss function was set to the cross-entropy loss. For the pdADMM-G-Q algorithm, $\Delta = \{-1, 0, 1, \cdots, 20\}$ in Problem 7, and $\mathbf{p}$ was quantized by default.

5.5.2 Comparison Methods

GD and its variants are state-of-the-art optimizers and hence served as comparison methods. For GD-based methods, all datasets were used for training models in a full-batch fashion. All hyperparameters were chosen by maximizing the performance of validation sets. Due to space limit, hyperparameter settings of all methods are shown in Section D.2.1 in the Appendix. The following are their brief introductions:

1. Gradient Descent(GD) [4]. The GD and its variants are the most popular deep learning optimizers. The GD updates parameters simply based on their gradients.

2. Adaptive learning rate method(Adadelta) [118]. The Adadelta is proposed to overcome the sensitivity to hyperparameter selection.

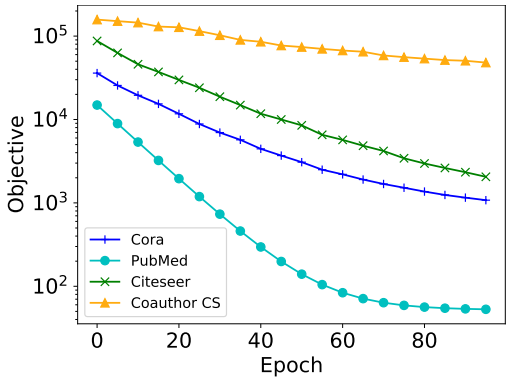
3. Adaptive gradient algorithm(Adagrad) [28]. Adagrad is an improved version of GD: rather than fixing the learning rate during training, it adapts the learning rate to the hyperparameter.

4. Adaptive momentum estimation(Adam) [52]. Adam is the most popular optimization method for deep learning models. It estimates the first and second momentum in order to correct the biased gradient, and thus accelerates empirical convergence.

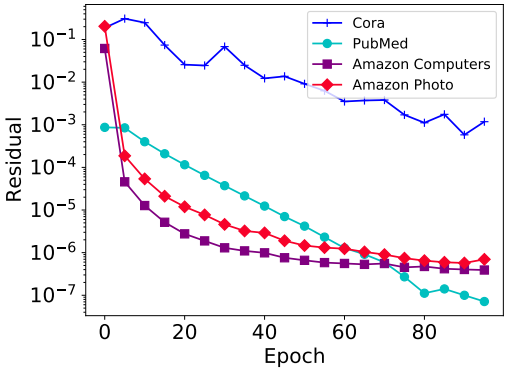
5.5.3 Convergence

Firstly, in order to validate the convergence of two proposed algorithms, we set up a GA-MLP model with 10 layers, each of which has 1000 neurons. The number of epochs was set to 100. ν and ρ were set to 0.01 and 1, respectively.

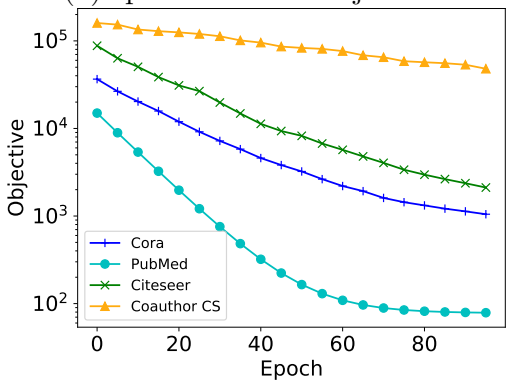
Figure 5.2 demonstrates objectives and residuals of two proposed algorithms on



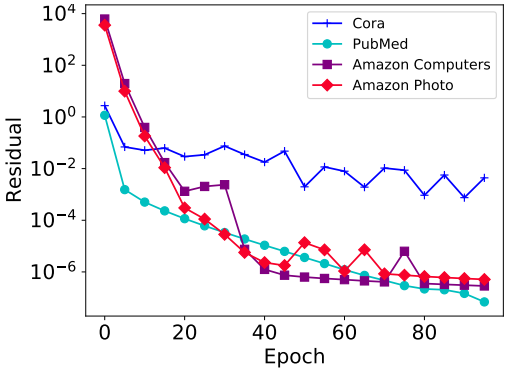
(a). pdADMM-G Objective.



(b). pdADMM-G Residual.

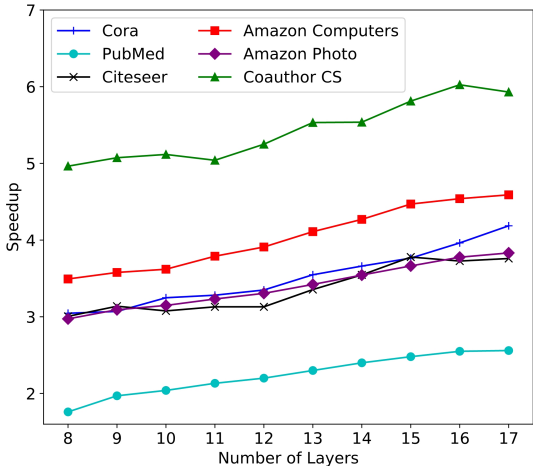


(c). pdADMM-G-Q Objective.

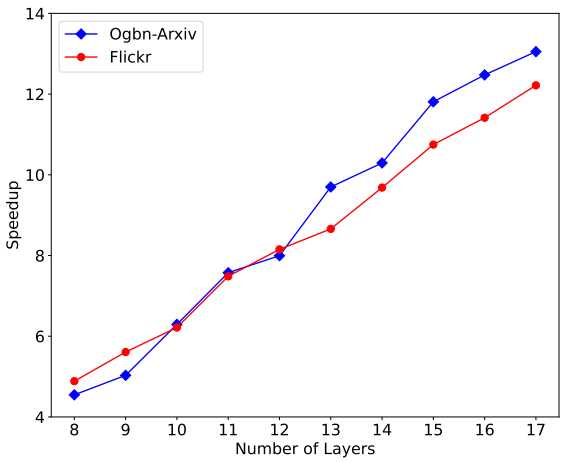


(d). pdADMM-G-Q Residual.

Figure 5.2: Convergence curves of the pdADMM-G algorithm and the pdADMM-G-Q algorithm on four datasets.



(a). The speedup on small datasets.



(b). The speedup on large datasets.

Figure 5.3: The speedup of the proposed pdADMM-G on different datasets concerning the number of layers(i.e. a weak-scaling study).

four datasets. Overall, the objectives and residuals of the two proposed algorithms are convergent. From Figure 5.2(a) and Figure 5.2(c), the objectives of the two proposed algorithms decrease drastically at the first 50 epochs and then drop smoothly to the end. The objectives on the PubMed dataset achieve the lowest among all four datasets, whereas these on the Coauthor CS dataset are the highest, which still reach near 10^5 at the 100-th epoch. As for residuals, even though the residuals of the pdADMM-G-Q algorithm are higher than these of the pdADMM-G algorithm initially, they both converge sublinearly to 0, which is consistent with Theorem 10 and Theorem 11. Specifically, as shown in Figure 5.2(b) and Figure 5.2(d), the residuals on the Cora dataset decrease more slowly with fluctuation than these on other datasets, while residuals on the Amazon Computers and Amazon Photo datasets demonstrate the fastest decreasing speed at the first 40 epochs before reaching a plateau less than 10^{-6}. The residuals on the PubMed dataset accomplish the lowest values among all four datasets again with a value of less than 10^{-7}.

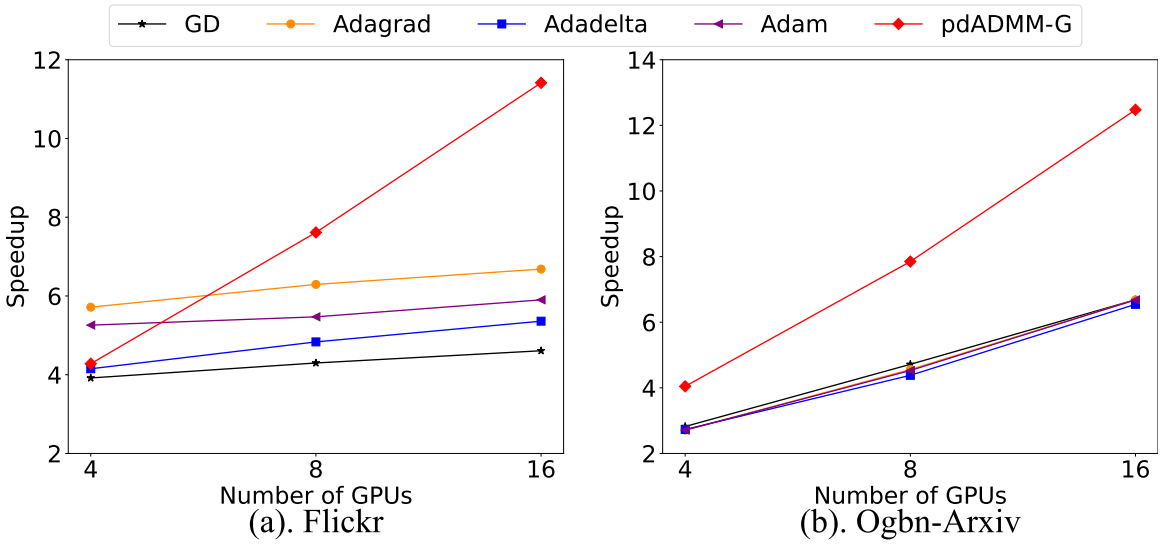


Figure 5.4: The speedup of all methods on two large datasets concerning the number of GPUs(i.e. a strong-scaling study).

5.5.4 Speedup

Next, we investigate the speedup of the pdADMM-G algorithm in the large deep GA-MLP models. The running time per epoch was an average of 10 epochs. ρ and ν were both set to 10^{-3}. We investigate the speedup concerning two factors: the number of layers(i.e. weak-scaling study) and the number of GPUs(i.e. strong-scaling study).

For the relationship between the speedup and the number of layers, the pdADMM-G algorithm in the GA-MLP models with 4000 neurons was tested. The number of layers ranged from 8 to 17. The speedup on small datasets and large datasets are shown in Figure 5.3(a) and Figure 5.3(b), respectively. Overall, the speedup of the proposed pdADMM-G increases linearly with the number of layers. For example, the speedups on the Cora dataset and the Amazon Computers dataset rise from 3 and 3.5 gradually to 4 and 4.5, respectively. The speedup on the PubMed dataset achieves the lowest with a value of less than 3, whereas that on the Coauthor CS dataset at least doubles that on any other small dataset, with a peak of 6. Moreover, the speedup is more obvious on large datasets. For example, when the slopes of speedups are compared, the slope on the Flickr dataset is at least five times much steeper than

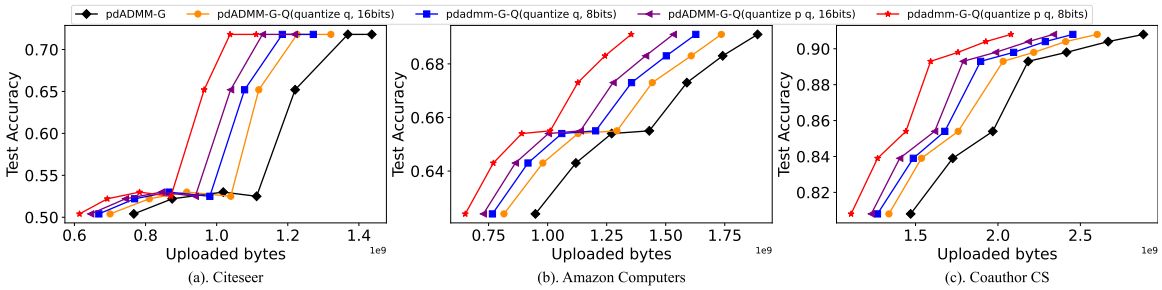


Figure 5.5: Communication overheads of two proposed algorithms on three datasets.

that on the Coauthor CS dataset. The same trend is applied to the Ogbn-Arxiv dataset. This means that our proposed pdADMM-G algorithm is more suitable for large datasets.

For the relationship between the speedup and the number of GPUs, we set up a large GA-MLP model with 16 layers and 4000 neurons and kept all hyperparameters in the previous experiment. The speedup of our proposed pdADMM-G algorithm was compared with all comparison methods. Figure 5.4 shows all speedups on two large datasets. The proposed pdADMM-G algorithm achieves a higher speedup than any GD-based method. For example, the speedups of 8 GPUs are nearly 8 on the Flickr dataset and the Ogbn-Arxiv dataset, while the best speedups achieved via comparison methods are in the vicinity of 6 and 5 on two datasets, respectively. We also observe that while speedups of all methods scale linearly with the number of GPUs, the slopes of our proposed pdADMM-G algorithm are steeper than these of any comparison method. For example, the slope of our proposed pdADMM-G algorithm on the Flickr dataset is more than 10 times steeper than that of Adam. All comparison methods show similar flat slopes, and they achieve a higher slope of the speedup on the Ogbn-Arxiv dataset than that on the Flickr dataset.

In summary, the speedup of our proposed pdADMM-G algorithm scales linearly with the number of layers and the number of GPUs. Moreover, its speedup is superior to any other comparison method significantly by more than 10 times.

5.5.5 Communication Overheads

Then, it is necessary to explore how many communication overheads can be reduced using the proposed quantization technique on different quantization levels. To achieve this, we established a large GA-MLP model with 10 layers, each of which consists of 1000 neurons. We set up three quantization cases: no quantization, the quantization concerning $\mathbf{p}$ only, and the quantization concerning both $\mathbf{p}$ and $\mathbf{q}$. For every quantization case, we also set up two different quantization sizes: 8 bits and 16 bits. Figure 5.5 demonstrates the relationship between the test accuracy and communication overheads for different quantization cases and sizes on three datasets. Overall, communication overheads can be reduced significantly by the proposed quantization technique. The amount of reduction depends on different quantization cases and sizes. Generally speaking, the more variables are quantized and the fewer bits are compressed, then the more savings in communications can be achieved. Take the Citeseer dataset as an example, while all algorithms reach the same test accuracy above 70%, the proposed pdADMM-G(i.e. no quantization) consumes the most communication costs with the value of around 1.4×10^9 bytes. If the variable $\mathbf{p}$ is quantized using 16 bits, the communication overhead drops by 10%, and then using 8 bits saves another 5%. When variables $\mathbf{p}$ and $\mathbf{q}$ are both quantized, the communication overhead tumbles down to 1.2×10^9 bytes, which means decreases by 16.7% when it is compared with the case where only $\mathbf{p}$ is quantized. When variables are compressed to 8 bits instead of 16 bits, the communication overhead slips further to 1.1×10^9, a nearly 30% decline. The same trend is applied to the other two datasets, and they accomplish a shrink of communication overheads by 33% and 45%, respectively. This demonstrates that our proposed quantization technique is effective for reducing unnecessary communication costs without loss of performance. We also observe that the Coauthor CS dataset is the largest dataset among the three, and it accomplishes the greatest communication reduction.

5.5.6 Performance

Finally, we evaluate the performance of two proposed algorithms against all comparison methods on nine benchmark datasets. We set up two standard GA-MLP models with 10 layers but different neurons: the first model has 100 neurons, while the second model has 500 neurons. Following the greedy layerwise training strategy [3], we firstly trained a two-layer GA-MLP model, and then three more layers were added to training, and finally, all 10 layers were involved. The number of epochs was set to 200. We repeated all experiments five times and reported their means and the standard deviations. Due to space limit, hyperparameter settings and the performance of validation sets are shown in Section D.2.1 and D.2.2 in the Appendix, respectively.

Table 5.3 demonstrates the performance of all methods when the number of neurons is 100. In summary, the two proposed algorithms outperform all comparison methods slightly: they occupy the best algorithms on eight datasets out of the total nine datasets. For example, they both achieve 78% test accuracy on the Cora dataset, whereas the best comparison method is GD, which only reaches 73% test accuracy, and is at least 6% lower than the two proposed algorithms. As another example, two proposed algorithms accomplish 78% test accuracy on the PubMed dataset, 4% better than that achieved by Adagrad, whose performance is the best aside from the two proposed algorithms. The Citeseer dataset shows the largest performance gap between the two proposed algorithms and all comparison methods. Two proposed algorithms reach the level of 70% test accuracy, whereas all comparison methods fall in the vicinity of 60% test accuracy. In other words, the two proposed algorithms outperform all comparison methods by more than 10%. For two proposed algorithms, the proposed pdADMM-G algorithm outperforms marginally the proposed pdADMM-G-Q algorithm due to the quantization technique. Their largest performance gap is 5%, which is achieved on the Amazon Computers dataset. The Adam is the best compar-

Dataset	Cora	PubMed	Citeseer
GD	0.730±0.022	0.638± 0.080	0.637±0.040
Adadelta	0.671±0.064	0.705±0.038	0.620±0.016
Adagrad	0.726±0.025	0.753± 0.015	0.601±0.037
Adam	0.725±0.036	0.742±0.007	0.631±0.018
pdADMM-G	0.784±0.003	**0.784±0.004**	0.709±0.003
pdADMM-G-Q	**0.788±0.003**	0.782±0.003	**0.712± 0.001**
Dataset	Amazon Computers	Amazon Photo	Coauthor CS
GD	0.646±0.032	0.735± 0.169	0.884±0.010
Adadelta	0.136±0.060	0.369± 0.045	0.787±0.086
Adagrad	0.688±0.023	0.813± 0.018	0.887±0.007
Adam	0.724±0.010	0.855±0.009	0.883±0.009
pdADMM-G	**0.735±0.006**	**0.856±0.011**	**0.915±0.004**
pdADMM-G-Q	0.687± 0.054	0.832±0.010	0.914±0.003
Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
GD	0.909±0.007	0.466±0.007	0.361±0.063
Adadelta	0.915±0.014	0.461±0.008	0.523± 0.030
Adagrad	0.916±0.012	0.481±0.003	0.567±0.016
Adam	0.912±0.016	0.512 ±0.004	**0.674± 0.006**
pdADMM-G	**0.921±0.003**	**0.513±0.002**	0.647±0.002
pdADMM-G-Q	0.919±0.002	0.507±0.003	0.655±0.002

Table 5.3: Test performance of all methods when the number of neurons is 100.

Dataset	Cora	PubMed	Citeseer
GD	0.757±0.024	0.699±0.655	0.680±0.014
Adadelta	0.717±0.063	0.722±0.696	0.564±0.028
Adagrad	0.776±0.013	0.759±0.761	0.650 ±0.038
Adam	0.771±0.020	0.778±0.767	0.662 ±0.021
pdADMM-G	**0.786±0.005**	0.786±0.786	**0.713±0.007**
pdADMM-G-Q	**0.786±0.005**	**0.788±0.787**	0.712±0.005
Dataset	Amazon Computers	Amazon Photo	Coauthor CS
GD	0.707±0.012	0.817±0.005	0.906±0.005
Adadelta	0.243±0.063	0.380±0.069	0.880±0.011
Adagrad	**0.753±0.009**	0.866±0.007	0.911±0.003
Adam	0.739±0.022	**0.880± 0.016**	0.898±0.013
pdADMM-G	0.751±0.008	0.873±0.004	**0.920±0.002**
pdADMM-G-Q	0.748±0.004	0.865±0.007	0.919±0.003
Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
GD	0.917±0.004	0.466±0.001	0.436±0.042
Adadelta	0.917±0.004	0.462±0.001	0.584±0.031
Adagrad	0.914±0.004	0.487±0.005	0.630±0.016
Adam	0.914±0.002	**0.517±0.002**	**0.682±0.010**
pdADMM-G	**0.918±0.003**	0.515±0.002	0.655±0.001
pdADMM-G-Q	**0.918±0.002**	0.512±0.003	0.657±0.002

Table 5.4: Test performance of all methods when the number of neurons is 500.

ison method overall, and it serves as the best algorithm on the Ogbn-Arxiv dataset. The Adadelta performs the worst among all comparison methods, whose performance is significantly lower than any other method on several datasets such as the Amazon Computers dataset, the Amazon Photo dataset, and the Coauthor CS dataset. Last but not least, the standard deviations of all methods remain low, and this shows that they are robust to different initializations.

Table 5.4 shows the performance of all methods when the number of neurons is 500. In general, two proposed algorithms still reach a better performance than all comparison methods, but the gap is more narrow. For example, in Table 5.3, the proposed pdADMM-G algorithm achieves the best on the Amazon Computers

dataset. However, it is surpassed by Adagrad slightly in Table 5.4. We also observe that a GA-MLP model with 500 neurons performs better than that with 100 neurons, which are trained by the same algorithm. This makes sense since the wider a model is, the more expressiveness it is equipped with.

5.6 Conclusion

The GA-MLP models are attractive to the deep learning community due to potential resistance to some problems of GNNs such as over-smoothing and over-squashing. In this chapter, we propose a novel pdADMM-G algorithm to achieve parallel training of GA-MLP models, which is accomplished by breaking the layer dependency. The extended pdADMM-G-Q algorithm reduces communication overheads by the introduction of the quantization technique. Their theoretical convergence to a(quantized) stationary point of the problem is guaranteed with a sublinear convergence rate $o(1/k)$, where k is the number of iterations. Extensive experiments verify that the two proposed algorithms not only converge in terms of objectives and residuals, and accelerate the training of deep GA-MLP models, but also stand out among all the existing state-of-the-art optimizers on nine benchmark datasets. Moreover, the pdADMM-G-Q algorithm reduces communication overheads by up to 45% without loss of performance.

Chapter 6

Conclusion and Future Works

The goal of this dissertation is to develop AM methods to train deep learning models as an alternative to GD. We focus on three research topics: In the topic of AM methods on Multi-Layer Perceptron(MLP), we devise two algorithms: deep learning Alternating Direction Method of Multipliers(dlADMM) and monotonous Deep Learning Alternating Minimization(mDLAM). In the topic of AM methods on Graph Neural Network(GNN), we propose the Invertible Validity-aware Graph Diffusion(IVGD) for graph source localization. In the topic of AM methods for distributed neural network training, we propose three algorithms: parallel deep learning Alternating Direction Method of Multipliers(pdADMM), graph pdADMM(pdADMM-G), and quantized graph pdADMM(pdADMM-G-Q).

For the dlADMM algorithm, the parameters in each layer are updated backward and then forward so that the parameter information in each layer is exchanged efficiently. The time complexity is reduced from cubic to quadratic in(latent) feature dimensions via a dedicated algorithm design for subproblems that enhances them by utilizing iterative quadratic approximations and backtracking. Finally, we provide the first proof of global convergence for an ADMM-based method(dlADMM) in a deep neural network problem under mild conditions. Experiments on benchmark

datasets demonstrated that our proposed dlADMM algorithm outperforms most of the comparison methods. Future work will further extend this method from Multi-Layer Perceptron(MLP) models to other architectures such as Graph Convolutional Network(GCN) models.

For the mDLAM algorithm, our innovative inequality-constrained formulation infinitely approximates the original problem with non-convex equality constraints, enabling our convergence proof of the proposed mDLAM algorithm regardless of the choice of hyperparameters. Our proposed mDLAM algorithm is shown to achieve a fast linear convergence by the Nesterov acceleration technique. Extensive experiments on multiple benchmark datasets demonstrate the convergence, effectiveness, and efficiency of the proposed mDLAM algorithm.

For the IVGD model, we make a graph diffusion model invertible by restricting its Lipschitz constant for the residual GNNs, and thus an approximate estimation of source localization can be obtained by its inversion, and then a compensation module is presented to reduce the introduced errors with skip connection. Moreover, we leverage the unrolled optimization technique to integrate validity constraints into the model, where each layer is encoded by a constrained optimization problem. To combat efficiency and scalability problems, a linearization technique is used to transform problems into solvable ones, which can be efficiently solved by closed-form solutions. Finally, the convergence of the proposed IVGD to a feasible solution is proven theoretically. Extensive experiments on nine datasets show that our proposed IVGD outperforms all comparison methods significantly on five metrics, especially 20% on F1-Score.

For the pdADMM algorithm, we achieve model parallelism training by breaking layer dependency among variables: parameters in each layer of neural networks can be updated independently in parallel. The convergence of the proposed pdADMM to a stationary point is theoretically proven under mild conditions. The convergence

rate of the pdADMM is proven to be $o(1/k)$, where k is the number of iterations. Extensive experiments on six benchmark datasets demonstrated that our proposed pdADMM can lead to more than a 10 times speedup for training large-scale deep neural networks, and outperformed most of the comparison methods.

For the pdADMM-G algorithm and the pdADMM-G-Q algorithm, they are extended from the proposed pdADMM algorithm to train GA-MLP models in parallel: the pdADMM-G algorithm breaks layer dependency similar to the pdADMM algorithm. The extended pdADMM-G-Q algorithm reduces communication costs by introducing the quantization technique. Theoretical convergence to a(quantized) stationary point of the pdADMM-G algorithm and the pdADMM-G-Q algorithm is provided with a sublinear convergence rate $o(1/k)$, where k is the number of iterations. Extensive experiments demonstrate the convergence of two proposed algorithms. Moreover, they lead to a more massive speedup and better performance than all state-of-the-art comparison methods on nine benchmark datasets. Last but not least, the proposed pdADMM-G-Q algorithm reduces communication overheads by up to 45% without loss of performance.

6.1 Research Tasks

The major research tasks are described as follows. The current status of these tasks is listed in Table 6.1.

6.1.1 The dlADMM Algorithm

- **Definition of the MLP training problem and problem transformation(A1).** We mathematically formulate the MLP training problem, and then transform the nonlinear constraints of this problem to ℓ_2 penalties in the objective.

Task	Description	Status
Research Work A	The dlADMM Algorithm	Completed
A1	Problem definition and transformation	Completed
A2	Proposal of the dlADMM algorithm	Completed
A3	Discussion of solutions to all subproblems	Completed
A4	Convergence proof of the proposed dlADMM algorithm	Completed
A5	Validation on the benchmark datasets	Completed
Research Work B	The mDLAM Algorithm	Completed
B1	Problem transformation and relaxation	Completed
B2	Proposal of the mDLAM algorithm	Completed
B3	Convergence proof of the proposed DLAM algorithm	Completed
B4	Validation on the benchmark datasets	Completed
Research Work C	The pdADMM Algorithm	Completed
C1	Problem definition and transformation	Completed
C2	Proposal of the pdADMM algorithm	Completed
C3	Convergence proof of the proposed pdADMM algorithm	Completed
C4	Validation on the benchmark datasets	Completed
Research Work D	The pdADMM-G-Q Algorithm and the pdADMM-G-Q Algorithm	Completed
D1	Problem definition and transformation	Completed
D2	Proposal of the pdADMM-G algorithm and the pdADMM-G-Q algorithm	Completed
D3	Convergence proofs of two proposed algorithms	Completed
D4	Validation on the benchmark datasets	Completed
Research Work E	The IVGD Model	
E1	Definition of the graph source localization problem	Completed
E2	Proposal of invertible graph residual network	Completed
E3	Proposal of graph validity-aware layers	Completed
E4	Convergence proof of the proposed IVGD model	Completed
E5	Validation on the real-world datasets	Completed
F	Dissertation writing and revision	Completed

Table 6.1: Research tasks and status

- **Proposal of the dlADMM algorithm(A2).** We present a novel and efficient dlADMM algorithm to handle the MLP training problem. The proposed dlADMM updates parameters in a backward-forward fashion, in order to accelerate the convergence process.

- **Discussion of solutions to all subproblems(A3).** We discuss solutions to all subproblems generated by the proposed dlADMM algorithm. For some subproblems, we propose quadratic approximation and backtracking techniques to avoid matrix inversion as well as reduce computational costs for large-scale datasets. The time complexity of subproblems in the proposed dlADMM is reduced from $O(n^3)$ to $O(n^2)$.

- **Convergence proof of the proposed dlADMM algorithm(A4).** We investigate several convergence properties of the proposed dlADMM. The convergence assumptions are very mild to ensure that most deep learning applications satisfy our assumptions. The proposed dlADMM is guaranteed to converge to a stationary point sublinearly whatever the initialization is when the hyperparameter is sufficiently large.

- **Validation on the benchmark datasets(A5).** We conduct experiments on several benchmark datasets to validate our proposed dlADMM algorithm. Experimental results show that the proposed dlADMM algorithm performs better than most GD-based optimizers. Moreover, we also demonstrate the convergence and the efficiency of the proposed dlADMM algorithm.

6.1.2 The mDLAM Algorithm

- **Definition of the MLP training problem and problem relaxation(B1).** We transform the original MLP training problem into an inequality-constrained problem that can infinitely approximate the original one. Applying this inno-

vation to an inequality-constraint-based transformation not only ensures the convexity of all subproblems but also reduces sensitivity to the input.

- **Proposal of the mDLAM algorithm(B2)**. We present an efficient mDLAM algorithm. The Nesterov acceleration technique is applied to further boost convergence. The quadratic approximation technique is utilized to avoid matrix inversion, and all subproblems have closed-form solutions.

- **Convergence proof of the proposed DLAM algorithm(B3).** We investigate several convergence properties of the proposed mDLAM algorithm under mild conditions. It is guaranteed to converge to a stationary point with linear convergence whatever hyperparameters are initialized. The convergence rate and the hyperparameter conditions of the proposed mDLAM algorithm are better than these of the previously proposed dlADMM algorithm.

- **Validation on the benchmark datasets(B4).** Extensive experiments on four benchmark datasets have been conducted to demonstrate the convergence, effectiveness, and scalability of the proposed mDLAM algorithm. It achieves a linear convergence as expected, and outperforms consistently state-of-the-art optimizers.

6.1.3 The pdADMM Algorithm

- **Definition of the MLP training problem and problem transformation(C1).** We propose a novel transformation of the MLP training problem by breaking the layer dependency, which splits a neural network into independent layer partitions and allows for the ADMM to achieve parallel training.

- **Proposal of the pdADMM algorithm(C2).** We propose a novel pdADMM algorithm to train the deep MLP model in parallel. All parameters in each layer

can be updated independently. All subproblems generated by the proposed pdADMM algorithm have closed-form solutions.

- **Convergence proof of the proposed pdADMM algorithm(C3).** Similar to the previously proposed dlADMM algorithm, we investigate the convergence properties of the proposed pdADMM algorithm under convergence assumptions, which are milder than these of the previously proposed dlADMM algorithm. We prove that it converges to a stationary point with sublinear convergence when the hyperparameters are sufficiently large.

- **Validation on the benchmark datasets(C4).** We conduct extensive experiments on six benchmark datasets to show the massive speedup of the proposed pdADMM as well as its competitive performance with state-of-the-art optimizers. The speedup of the proposed pdADMM algorithm increases linearly with the number of layers.

6.1.4 The pdADMM-G and pdADMM-G-Q Algorithms

- **Definition of the GA-MLP training problem and problem transformation(D1).** We mathematically formulate the GA-MLP training problem, and then break layer dependency to achieve parallel training, which is similar to the previously proposed pdADMM algorithm.

- **Proposal of the pdADMM-G algorithm and the pdADMM-G-Q algorithm(D2).** We extend the previously proposed pdADMM algorithm to train the GA-MLP model, named the pdADMM-G algorithm. The extended pdADMM-G-Q algorithm reduces communication costs by introducing the quantization technique.

- **Convergence proofs of two proposed algorithms(D3).** We provide the theoretical convergence guarantees of two proposed algorithms. Specifically, we

investigate their several convergence properties and prove that they converge to a(quantized) stationary point of GA-MLP models sublinearly when the hyper-parameters are sufficiently large.

- **Validation on the benchmark datasets(D4).** We conduct extensive experiments on nine benchmark datasets to show the convergence, the massive speedup of two proposed algorithms, as well as their outstanding performance. Moreover, the proposed pdADMM-G-Q algorithm reduces communication overheads significantly by up to 45%.

6.1.5 The IVGD Model

- **Definition of the graph source localization problem(E1).** We define the graph source localization problem where the prior knowledge of the graph diffusion model is available. Our goal is to design a graph source localization model constrained by graph validity constraints via utilizing such prior knowledge.

- **Proposal of invertible graph residual network(E2).** We propose a new graph residual network with Lipschitz regularization to ensure the invertibility of graph diffusion models. Furthermore, we propose an error compensation mechanism to offset the errors inferred from the graph residual network.

- **Proposal of graph validity-aware layers(E3).** We ensure the validity of inferred sources by automatically learning validity-aware layers. We further accelerate the optimization of the proposed layers by leveraging a linearization technique. It transforms nonconvex problems into convex problems, which have closed-form solutions.

- **Convergence proof of the proposed IVGD model(E4).** We prove the convergence of the proposed IVGD model for linear validity constraints. It con-

verges asymptotically to a feasible solution when the number of graph validity-aware layers goes to infinity.

- **Validation on the real-world datasets(E5).** Extensive experiments on nine datasets have been conducted to demonstrate the effectiveness and robustness of our proposed IVGD. Our proposed IVGD outperforms all comparison methods significantly on five metrics.

6.2 Discussion

Finally, we summarize some recent training methods in parallel to our publications. These training methods can be either serial training methods or parallel training methods. Serial training methods are classified into two categories: 1). Stochastic Approximation(SA). This type of method trains a neural network in a stochastic manner. For example, Newman et al. proposed an algorithm called slimTrain, which automatically adapted regularization parameters for linear weights [74]. 2). Sample Average Approximation(SAA). SAA methods update the parameters of neural networks in a full-batch fashion, where our proposed AM methods fall. As an example, Newman et al. proposed the GNvpro by extending the idea of variable projection in the nonlinear least-square problem to train Deep Neural Network(DNN) training [73]. The main difference between our proposed AM methods and theirs consists in the separability of neural networks: slimTrain and GNvpro split a neural network into a nonlinear feature extractor followed by an affine mapping, while our proposed AM methods have a finer granularity of decomposition: they decompose a neural network into layerwise training modules. Such difference leads to several advantages: firstly, our proposed AM methods are proven to converge for non-differentiable activations(e.g. ReLU) [101, 106], while no such convergence guarantees are provided for slimTrain and GNvpro; secondly, they can avoid gradient vanishing/exploding prob-

lems for any neural network architecture [93, 92], whereas slimTrain and GNvpro require delicate designs of neural networks (e.g. hyperbolic neural networks); thirdly, these training modules usually have analytic solutions and hence facilitate efficient training, and our proposed AM methods converge to modest accuracy within tens of iterations, which are sufficient for real-world applications [6].

As for parallel training methods, they can be categorized into data parallelism and model parallelism. We have summarized their recent development in Chapters 4 and 5. One important related work is an MGRIT-based method proposed by Gunther et al. [32], as their method is also layerwise-based. Specifically, they substitute a parallel nonlinear multigrid iteration applied to the layer domain for the backpropagation algorithm. While their mechanism is different from our proposed AM methods, they share a common characteristic: they both forsake model precision for speedup: the MGRIT-based method employs a relaxation scheme to approximate the true solution on the fine grid; our proposed AM methods relax original training problems via imposing constraints as penalties on the objectives. One potential future direction is to combine their idea with ours to accelerate model training further.

It is important to compare our proposed AM methods with Block Coordinate Descent(BCD) methods. This is because they are important components of SAA methods and share some commonalities with our AM methods, which have been investigated recently to train neural networks [19, 120, 124]. Please refer to Chapter 3 for recent development. While they have achieved excellent performance on many datasets, their usage is restricted by the challenges of the ill-conditioning and coupled variables [73, 74]. For our proposed AM methods in this dissertation, they solve subproblems approximately: we replace each subproblem with a quadratic approximation, and optimal learning parameters are selected via iterative backtracking, and therefore, we can choose accurate directions toward solutions.

6.3 Current Publications

6.3.1 Contributions of Published Papers Contributing to dissertation

The following is a list of published papers that contribute to the dissertation and author contributions:

1. Junxiang Wang, Fuxun Yu, Xiang Chen, and Liang Zhao. ADMM for Efficient Deep Learning with Global Convergence. in Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining(KDD 2019), research track(acceptance rate: 14.2%), Alaska, USA, Aug 2019.

   Junxiang Wang proposed the dlADMM algorithm, proved the convergence of the dlADMM algorithm, led the experimental design and analysis and drafted the manuscript. Fuxun Yu participated in the experimental design, and Xiang Chen and Liang Zhao provided support and manuscript editing.

2. Junxiang Wang, Hongyi Li, and Liang Zhao. Accelerated Gradient-free Neural Network Training by Multi-convex Alternating Optimization. Neurocomputing,(impact factor: 5.719), 2022.

   Junxiang Wang proposed the mDLAM algorithm, proved the convergence of the mDLAM algorithm, led the experimental design and analysis and drafted the manuscript. Hongyi Li participated in the experimental design, and Liang Zhao provided support and manuscript editing.

3. Junxiang Wang, Zheng Chai, Yue Cheng, Liang Zhao. Toward Model Parallelism for Deep Neural Network based on Gradient-free ADMM Framework. in Proceedings of the IEEE International Conference on Data Mining(ICDM 2020), regular paper(acceptance rate: 9.8%), Sorrento, Italy, Nov 2020.

   Junxiang Wang proposed the pdADMM algorithm, proved the convergence of

the pdADMM algorithm, led the experimental design and analysis and drafted the manuscript. Zheng Chai participated in the experimental design, and Yue Cheng and Liang Zhao provided support and manuscript editing.

4. Junxiang Wang, Hongyi Li, Zheng Chai, Yongchao Wang, Yue Cheng, and Liang Zhao. Towards Quantized Model Parallelism for Graph-Augmented MLPs Based on Gradient-Free ADMM Framework, IEEE Transactions on Neural Networks and Learning Systems(TNNLS),(impact factor: 14.425).
   Junxiang Wang proposed the pdADMM-G algorithm and pdADMM-G-Q algorithm, proved the convergence of the two proposed algorithms, participated in the experimental design and drafted the manuscript. Hongyi Li led the experimental design and analysis. Zheng Chai participated in the experimental design. Yongchao Wang, Yue Cheng, and Liang Zhao provided support and manuscript editing.

5. Junxiang Wang, Junji Jiang, and Liang Zhao. An Invertible Graph Diffusion Neural Network for Source Localization. 31st International World Wide Web Conference (WWW 2022), (acceptance rate: 17.7%), Lyon, FR, Apr 2022.
   Junxiang Wang proposed the IVGD framework, proved the convergence of the proposed IVGD, led the experimental design and analysis and drafted the manuscript. Junji Jiang participated in the experimental design. Liang Zhao provided support and manuscript editing.

6.3.2 Published Papers During My Ph.D.

Conference Papers

1. Chen Ling, Tanmoy Chowdhury, Junji Jiang, Junxiang Wang, Xuchao Zhang, Haifeng Chen, and Liang Zhao. DeepAR: Deep Graph Representation Learning and Optimization for Analogical Reasoning. in Proceedings of the IEEE

International Conference on Data Mining(ICDM 2022), short paper(acceptance rate: 20%), Orlando, FL, USA, Nov 2022.

2. Chen Ling, Junji Jiang, Junxiang Wang, and Liang Zhao. SL-VAE: Variational Autoencoder for Source Localization. in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining(KDD 2022), research track(acceptance rate: 15.0%), Washington D.C, USA, Aug 2022.

3. Junxiang Wang, Junji Jiang, and Liang Zhao. An Invertible Graph Diffusion Neural Network for Source Localization. 31th International World Wide Web Conference(WWW 2022),(acceptance rate: 17.7%), Lyon, FR, Apr 2022.

4. Junxiang Wang and Liang Zhao. Convergence and Applications of ADMM on the Multi-convex Problems. 26th Pacific-Asia Conference on Knowledge Discovery and Data Mining(PAKDD 2022),(acceptance rate: 19.3%), Chengdu, China, May 2022.

5. Junxiang Wang, Zheng Chai, Yue Cheng, Liang Zhao. Toward Model Parallelism for Deep Neural Network based on Gradient-free ADMM Framework. in Proceedings of the IEEE International Conference on Data Mining(ICDM 2020), regular paper(acceptance rate: 9.8%), Sorrento, Italy, Nov 2020.

6. Junxiang Wang, Fuxun Yu, Xiang Chen, and Liang Zhao. ADMM for Efficient Deep Learning with Global Convergence. in Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining(KDD 2019), research track(acceptance rate: 14.2%), Alaska, USA, Aug 2019.

7. Junxiang Wang, Liang Zhao, and Yanfang Ye. Semi-supervised Multi-instance Interpretable Models for Flu Shot Adverse Event Detection. 2018 IEEE International Conference on Big Data(BigData 2018)(acceptance rate: 18.9%),

Seattle, USA, Dec 2018.

8. Junxiang Wang, Yuyang Gao, Andreas Zufle, Jingyuan Yang, and Liang Zhao. Incomplete Label Uncertainty Estimation for Petition Victory Prediction with Dynamic Features. in Proceedings of the IEEE International Conference on Data Mining(ICDM 2018), regular paper(acceptance rate: 8.9%), Singapore, Dec 2018.

9. Junxiang Wang and Liang Zhao. Multi-instance Domain Adaptation for Vaccine Adverse Event Detection. 27th International World Wide Web Conference(WWW 2018),(acceptance rate: 14.8%), Lyon, FR, Apr 2018.

10. Liang Zhao, Junxiang Wang, and Xiaojie Guo. Distant-supervision of heterogeneous multitask learning for social event forecasting with multilingual indicators. Thirty-Second AAAI Conference on Artificial Intelligence(AAAI 2018), Oral presentation(acceptance rate: 11.0%), pp. 4498-4505, New Orleans, US, Feb 2018.

Journal Papers

1. Junxiang Wang, Hongyi Li(first-coauthor), Zheng Chai, Yongchao Wang, Yue Cheng, and Liang Zhao. Towards Quantized Model Parallelism for Graph-Augmented MLPs Based on Gradient-Free ADMM Framework. IEEE Transactions on Neural Networks and Learning Systems(TNNLS),(impact factor: 14.225).

2. Junxiang Wang, Hongyi Li, and Liang Zhao. Accelerated Gradient-free Neural Network Training by Multi-convex Alternating Optimization. Neurocomputing,(impact factor: 5.719), 2022.

3. Junxiang Wang and Liang Zhao. Nonconvex Generalization of Alternating Di-

rection Method of Multipliers for Nonlinear Equality Constrained Problems. Results in Control and Optimization 2: 100009, 2021.

4. Junxiang Wang, Liang Zhao, Yanfang Ye, and Yuji Zhang. Adverse event detection by integrating Twitter data and VAERS. Journal of Biomedical Semantics,(impact factor: 1.845), 2018.

5. Liang Zhao, Junxiang Wang, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. "Spatial Event Forecasting in Social Media with Geographically Hierarchical Regularization". Proceedings of the IEEE(impact factor: 9.237), vol. 105, no. 10, pp. 1953-1970, Oct. 2017.

6.3.3 Papers Published Before Ph.D.

1. Junxiang Wang, Weiming Yu, Zhibin Chen, Hengda Li, Zhenran Jiang. Predicting Drug-Target Interactions of Nuclear Receptors Based on Molecular Descriptors Information. Letters in Drug Design and Discovery 10(10), 989-994, 2013.

2. Weiming Yu, Yan Yan, Qing Liu, Junxiang Wang, and Zhenran Jiang. Predicting drug-target interaction networks of human diseases based on multiple feature information. Pharmacogenomics 14(14), 1701-1707, 20, 2013.

3. Zhenran Jiang, Ran Tao, Lei Du, Weiming Yu and Junxiang Wang. Using Network-Based Approaches to Predict Ligands of Orphan Nuclear Receptors. Current Bioinformatics 7(4), 411-414, 2012.

4. Ran Tao, Zhenran Jiang, Weiming Yu and Junxiang Wang. Predicting Coupling Specificity of GPCRs Based on the Optimization of the Coupling Regions. Combinatorial Chemistry and High Throughput Screening 15(9), 770-774, 2012.

5. Weiming Yu, Zhengyan Jiang, Junxiang Wang, and Ran Tao. Using feature selection technique for drug-target interaction networks prediction. Current medicinal chemistry 18(36), 5687-5693, 2011.

6.3.4 Submitted and In-preparation Papers

1. Junxiang Wang, Hongyi Li, and Liang Zhao. Proximal ADMM Algorithms for Multi-convex Problems. International Journal of Data Science and Analytics(Submission)

2. Junxiang Wang, Junji Jiang, and Liang Zhao. An Invertible Bi-Lipschitz Surrogate Model for Black-box Graph Inverse Problems. NeurIPS 2022(Submission)

3. Hongyi Li, Junxiang Wang, Yongchao Wang, Yue Cheng, and Liang Zhao. Community-based Layerwise Distributed Training of Graph Convolutional Networks. IEEE Transactions on Neural Networks and Learning Systems(TNNLS) (Submission)

4. Chen Ling, Junji Jiang, Junxiang Wang, Renhao Xue, and Zhao Liang. DeepIM: Deep Graph Representation Learning and Optimization for Influence Maximization. WSDM 2023(In Preparation)

5. Junxiang Wang, Hongyi Li, Liang Zhao. A Convergent ADMM Framework for Efficient Neural Network Training. IEEE Transactions on Knowledge and Data Engineering(TKDE)(In Preparation)

6. Yuyang Gao, Junxiang Wang, Wei Wang, Xin Deng, Hamed Zamani, Xiaohan Yan, Yan Guo, Ahmed Awadallah, Yanfang Ye, and Liang Zhao. Asynchronous Semi-supervised Representation Learning for Email Heterogeneous Networks. CIKM 2022(Submission)

7. Johnny Torres, Guangji Bai, Junxiang Wang, Liang Zhao, Carmen Vaca, Cristina Abad. Sign-regularized Multi-task Learning. Neurocomputing(In Preparation)

6.4 Future Research Directions

6.4.1 Parallel Training of Graph Neural Networks on Large-Scale Graphs

The GCN is one of the most popular graph learning models: it demonstrates the ability to express nodes and graphs effectively via the message-passing mechanism. Despite its effectiveness, it is challenging to train large-scale GCN models. This is because the space complexity and the time complexity to handle graphs are quadratic with respect to their nodes. Therefore, a graph with millions or billions of nodes may not fit in memory. Potential future work is to devise AM algorithms to address parallel training on large-scale graphs.

6.4.2 Stochastic AM Algorithms for Large-Scale Datasets

On one hand, the full-batch GD is popular in the small machine learning problems, they suffer from memory issues on the large-scale datasets, which is also potentially a challenge of SSA methods, including our proposed AM methods. On the other hand, however, stochastic algorithms such as Stochastic Gradient Descent(SGD) are more prevalent for large-scale datasets [5]. This motivates us to extend our proposed AM methods to stochastic formats by some variance reduction techniques such as SVRG [50] and SAGA [24]. We will investigate their theoretical convergence rates as well as verify their performance on large-scale datasets.

Appendix A

Appendix of the dlADMM Algorithm

A.1 Algorithms to Update W_l^{k+1} and a_l^{k+1}

The algorithms to update W_l^{k+1} and a_l^{k+1} are described in the Algorithms 7 and 8, respectively.

Algorithm 7: The Backtracking Algorithm to Update W_l^{k+1}

Require: $\mathbf{W}_{l-1}^{k+1}$, $\mathbf{z}_{l-1}^{k+1}$,$\mathbf{a}_{l-1}^{k+1}$,u^k, ρ, some constant $\gamma > 1$.
Ensure: θ_l^{k+1},W_l^{k+1}.

1: Pick up α and $\zeta = \overline{W}_l^{k+1} - \nabla_{\overline{W}_l^{k+1}}\phi/\alpha$.
2: **while** $\phi(\{W_i^{k+1}\}_{i=1}^{l-1}, \zeta, \{\overline{W}_i^{k+1}\}_{i=l+1}^{L}, \mathbf{z}_l^{k+1}, \mathbf{a}_l^{k+1}, u^k) > P_l(\zeta; \alpha)$ **do**
3: $\quad \alpha \leftarrow \alpha\ \gamma$.
4: $\quad$ Solve ζ by Equation (2.9).
5: **end while**
6: Output $\theta_l^{k+1} \leftarrow \alpha$.
7: Output $W_l^{k+1} \leftarrow \zeta$.

A.2 Preliminary Lemmas

The following several lemmas are preliminary results.

Algorithm 8: The Backtracking Algorithm to Update a_l^{k+1}

Require: $\mathbf{W}_l^{k+1}$, $\mathbf{z}_l^{k+1}$,$\mathbf{a}_{l-1}^{k+1}$,u^k, ρ, some constant $\eta > 1$.
Ensure: τ_l^{k+1},a_l^{k+1}.
1: Pick up t and $\beta = \overline{a}_l^{k+1} - \nabla_{\overline{a}_l^{k+1}} \phi / t$
2: **while** $\phi(\mathbf{W}_{l+1}^{k+1}, \mathbf{z}_{l+1}^{k+1}, \{a_i^{k+1}\}_{i=1}^{l-1}, \beta, \{\overline{a}_i^{k+1}\}_{i=l+1}^{L-1}, u^k) > Q_l(\beta; t)$ **do**
3: $t \leftarrow t\eta$.
4: $\beta \leftarrow \overline{a}_l^{k+1} - \nabla_{\overline{a}_l^{k+1}} \phi / t$.
5: **end while**
6: Output $\tau_l^{k+1} \leftarrow t$.
7: Output $a_l^{k+1} \leftarrow \beta$.

Lemma 11. $\nabla_{z_L^k} R(z_L^k; y) + u^k = 0$ for all $k \in \mathbb{N}$.

Proof. The optimality condition of z_L^k in Equation (2.11) gives rise to

$$\nabla_{z_L^k} R(z_L^k; y) + \rho(z_L^k - W_L^k a_{L-1}^k) + u^{k-1} = 0$$

Because $u^k = u^{k-1} + \rho(z_L^k - W_L^k a_{L-1}^k)$, then we have $\nabla_{z_L^k} R(z_L^k; y) + u^k = 0$. □

Lemma 12. It holds that $\forall z_{L,1}, z_{L,2} \in \mathbb{R}^{n_L}$,

$$R(z_{L,1}; y) \leq R(z_{L,2}; y) + \nabla_{z_{L,2}} R^T(z_{L,2}; y)(z_{L,1} - z_{L,2}) + (H/2)\|z_{L,1} - z_{L,2}\|^2$$
$$- R(z_{L,1}; y) \leq -R(z_{L,2}; y) - \nabla_{z_{L,2}} R^T(z_{L,2}; y)(z_{L,1} - z_{L,2}) + (H/2)\|z_{L,1} - z_{L,2}\|^2$$

Proof. Because $R(z_L; y)$ is Lipschitz differentiable by Assumption 2, so is $-R(z_L; y)$. Therefore, this lemma is proven exactly as same as Lemma 2.1 in [2]. □

Lemma 13. It holds that for $\forall k \in \mathbb{N}$,

$$L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l+1}^{k+1}, u^k) - L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l}^{k+1}, u^k)$$
$$\geq \|\overline{\tau}_l^{k+1} \circ (\overline{a}_l^{k+1} - a_l^k)^{\circ 2}\|_1/2 (l = 1, \cdots, L-1) \tag{A.1}$$
$$L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l+1}^{k+1}, \overline{\mathbf{a}}_{l}^{k+1}, u^k) \geq L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l}^{k+1}, \overline{\mathbf{a}}_{l}^{k+1}, u^k) (l = 1, \cdots, L-1) \tag{A.2}$$
$$L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k) - L_\rho(\mathbf{W}^k, \overline{\mathbf{z}}_L^{k+1}, \mathbf{a}^k, u^k) \geq (\rho/2)\|\overline{z}_L^{k+1} - z_L^k\|_2^2 \tag{A.3}$$
$$L_\rho(\overline{\mathbf{W}}_{l+1}^{k+1}, \overline{\mathbf{z}}_{l}^{k+1}, \overline{\mathbf{a}}_{l}^{k+1}, u^k) - L_\rho(\overline{\mathbf{W}}_{l}^{k+1}, \overline{\mathbf{z}}_{l}^{k+1}, \overline{\mathbf{a}}_{l}^{k+1}, u^k)$$
$$\geq \|\overline{\theta}_l^{k+1} \circ (\overline{W}_l^{k+1} - W_l^k)^{\circ 2}\|_1/2 (l = 1, \cdots, L) \tag{A.4}$$
$$L_\rho(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) - L_\rho(\mathbf{W}_{l}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k)$$
$$\geq \|\theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1})^{\circ 2}\|_1/2 (l = 1, \cdots, L) \tag{A.5}$$
$$L_\rho(\mathbf{W}_{l}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \geq L_\rho(\mathbf{W}_{l}^{k+1}, \mathbf{z}_{l}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) (l = 1, \cdots, L-1) \tag{A.6}$$
$$L_\rho(\mathbf{W}_{l}^{k+1}, \mathbf{z}_{l}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) - L_\rho(\mathbf{W}_{l}^{k+1}, \mathbf{z}_{l}^{k+1}, \mathbf{a}_{l}^{k+1}, u^k)$$
$$\geq \|\tau_l^{k+1} \circ (a_l^{k+1} - \overline{a}_l^{k+1})^{\circ 2}\|_1/2 (l = 1, \cdots, L-1) \tag{A.7}$$

Proof. Essentially, all inequalities can be obtained by applying optimality conditions of updating $\overline{a}_l^{k+1}$, $\overline{z}_l^{k+1}$, $\overline{W}_l^{k+1}$, W_l^{k+1}, z_l^{k+1} and a_l^{k+1}, respectively. We only prove Inequality (A.3), (A.5), and (A.6). This is because Inequalities (A.1), (A.4) and (A.7) follow the routine of Inequality (A.5), and Inequality (A.2) follows the routine of Inequality (A.6).

Firstly, we focus on Inequality (A.3).

$$\begin{aligned}
&L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k) - L_\rho(\mathbf{W}^k, \overline{\mathbf{z}}_L^{k+1}, \mathbf{a}^k, u^k) \\
&= R(z_L^k; y) + (u^k)^T(z_L^k - W_L^k a_{L-1}^k) + (\rho/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \\
&- R(\overline{z}_L^{k+1}; y) - (u^k)^T(\overline{z}_L^{k+1} - W_L^k a_{L-1}^k) - (\rho/2)\|\overline{z}_L^{k+1} - W_L^k a_{L-1}^k\|_2^2 \\
&= R(z_L^k; y) - R(\overline{z}_L^{k+1}; y) + (u^k)^T(z_L^k - \overline{z}_L^{k+1}) + (\rho/2)\|\overline{z}_L^{k+1} - z_L^k\|_2^2 \\
&+ \rho(\overline{z}_L^{k+1} - W_L^k a_{L-1}^k)^T(z_L^k - \overline{z}_L^{k+1})
\end{aligned} \tag{A.8}$$

where the second equality follows from the cosine rule $\|z_L^k - W_L^k a_{L-1}^k\|_2^2 - \|\overline{z}_L^{k+1} - W_L^k a_{L-1}^k\|_2^2 = \|\overline{z}_L^{k+1} - z_L^k\|_2^2 + (\overline{z}_L^{k+1} - W_L^k a_{L-1}^k)^T(z_L^k - \overline{z}_L^{k+1})$.

According to the optimality condition of Equation (2.5), we have $\nabla_{\overline{z}_L^{k+1}} R(\overline{z}_L^{k+1}; y) + u^k + \rho(\overline{z}_L^{k+1} - W_L^k a_{L-1}^k) = 0$. Because $R(z_L; y)$ is convex and differentiable with regard to z_L, its subgradient is also its gradient. According to the definition of subgradient, we have

$$\begin{aligned} R(z_L^k; y) &\geq R(\overline{z}_L^{k+1}; y) + \nabla_{\overline{z}_L^{k+1}} R^T(\overline{z}_L^{k+1}; y)(z_L^k - \overline{z}_L^{k+1}) \\ &= R(\overline{z}_L^{k+1}; y) - (u^k + \rho(\overline{z}_L^{k+1} - W_L^k a_{L-1}^k))^T(z_L^k - \overline{z}_L^{k+1}) \end{aligned} \tag{A.9}$$

We introduce Equation (A.9) into Equation (A.8) to obtain Equation (A.3).

Secondly, we focus on Inequality (A.5). The choice of θ_l^{k+1} in Algorithm 7 shows that

$$\phi(\mathbf{W}_l^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \leq P_l(W_l^{k+1}; \theta_l^{k+1}). \tag{A.10}$$

Moreover, the optimality condition of W_l^{k+1} shows that

$$(\nabla_{\overline{W}_l^{k+1}} \phi)^T(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) + \theta_l^{k+1} \circ (W_l - W_l^k) = 0 \tag{A.11}$$

Therefore, we have

$$\begin{aligned} &L_\rho(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) - L_\rho(\mathbf{W}_l^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\ &= \phi(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) - \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \quad \text{(Definition of } L_\rho) \\ &\geq \phi(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) - P_l(W_l^{k+1}; \theta_l^{k+1}) \quad \text{(Equation (A.10))} \\ &= -\nabla \phi_{\overline{W}_l^{k+1}}^T (W_l^{k+1} - \overline{W}_l^{k+1}) - \|\theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1})^{\circ 2}\|_1/2 \\ &= \|\theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1})^{\circ 2}\|_1/2 \quad \text{(Equation (A.11)).} \end{aligned}$$

Finally, we focus on Equation (A.6). This follows directly from the optimality of z_l^{k+1} in Equation (2.10). □

Lemma 14. If $\rho > \frac{\sqrt{17}+1}{2}H$ so that $C_1 = \rho/2 - H/2 - 2H^2/\rho > 0$, then it holds that

$$\begin{aligned} &L_\rho(\mathbf{W}^{k+1}, \mathbf{z}_{L-1}^{k+1}, \mathbf{a}^{k+1}, u^k) - L_\rho(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\ &\geq C_1 \|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 - (2H^2/\rho)\|\overline{z}_L^{k+1} - z_L^k\|_2^2. \end{aligned} \tag{A.12}$$

Proof.

$$\begin{aligned} &L_\rho(\mathbf{W}^{k+1}, \mathbf{z}_{L-1}^{k+1}, \mathbf{a}^{k+1}, u^k) - L_\rho(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\ &= R(\overline{z}_L^{k+1}; y) - R(z_L^{k+1}; y) + (u^{k+1})^T(\overline{z}_L^{k+1} - z_L^{k+1}) + (\rho/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 \\ &- (1/\rho)\|u^{k+1} - u^k\|_2^2 \\ &= R(\overline{z}_L^{k+1}; y) - R(z_L^{k+1}; y) + \nabla_{z_L^{k+1}} R(z_L^{k+1}; y)^T(z_L^{k+1} - \overline{z}_L^{k+1}) + (\rho/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 \\ &- (1/\rho)\|u^{k+1} - u^k\|_2^2 (\text{ Lemma 11}) \\ &\geq (-H/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 + (\rho/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 - (1/\rho)\|\nabla_{z_L^{k+1}} R(z_L^{k+1}; y) \\ &- \nabla_{z_L^k} R(z_L^k; y)\|_2^2 (-R(z_L; y) \text{ is Lipschitz differentiable, Lemmas 11 and 12}) \\ &\geq (-H/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 + (\rho/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 - (H^2/\rho)\|z_L^{k+1} - z_L^k\|_2^2 \\ &(\text{Assumption 2}) \\ &\geq (-H/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 + (\rho/2)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 - (2H^2/\rho)\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 \\ &- (2H^2/\rho)\|\overline{z}_L^{k+1} - z_L^k\|_2^2 \quad (\text{Mean Inequality}) \\ &= C_1\|z_L^{k+1} - \overline{z}_L^{k+1}\|_2^2 - (2H^2/\rho)\|\overline{z}_L^{k+1} - z_L^k\|_2^2. \end{aligned}$$

We choose $\rho > \frac{\sqrt{17}+1}{2}H$ to make $C_1 > 0$. □

A.3 Proof of Theorem 1

Proving Theorem 1 is equal to proving jointly Theorem 13, 14, and 15, which are elaborated in the following.

Theorem 13. Given that Assumptions 1 and 2 hold, the dlADMM satisfies Property 1.

Proof. There exists $z_L^{'}$ such that $z_L^{'} - W_L^k a_{L-1}^k = 0$. By Assumption 2, we have

$$F(\mathbf{W}^k, \{z_l^k\}_{l=1}^{L-1}, z_L^{'}, \mathbf{a}^k) \geq \min S > -\infty$$

where $S = \{F(\mathbf{W}, \mathbf{z}, \mathbf{a}) : z_L - W_L a_{L-1} = 0\}$. Then we have

$$\begin{aligned}
&L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k) \\
&= F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) + (u^k)^T(z_L^k - W_L^k a_{L-1}^k) + (\rho/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \\
&= F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) + (u^k)^T(z_L^k - z_L^{'}) + (\rho/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \\
&(z_L^{'} - W_L^k a_{L-1}^k = 0) \\
&= F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) + \nabla_{z_L^k} R^T(z_L^k; y)(z_L^{'} - z_L^k) + (\rho/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \\
&\text{(Lemma 11)} \\
&= (\nu/2)\sum_{l=1}^{L-1}(\|z_l^k - W_l^k a_{l-1}^k\|_2^2 + \|a_l^k - f(z_l^k)\|_2^2) + R(z_L^k; y) \\
&+ \nabla_{z_L^k} R^T(z_L^k; y)(z_L^{'} - z_L^k) + (\rho/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \quad \text{(The definition of } F\text{)} \\
&\geq (\nu/2)\sum_{l=1}^{L-1}(\|z_l^k - W_l^k a_{l-1}^k\|_2^2 + \|a_l^k - f(z_l^k)\|_2^2) + R(z_L^{'}; y) \\
&+ (\rho - H/2)\|z_L^k - W_L^k a_{L-1}^k\|_2^2 \\
&\text{(Lemmas 11 and 12, } R(z_L; y) \text{ is Lipschitz differentiable)} \\
&> -\infty
\end{aligned}$$

It concludes from Lemma 13 and Lemma 14 that $L_\rho(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k, u^k)$ is upper bounded by $L_\rho(\mathbf{W}^0, \mathbf{z}^0, \mathbf{a}^0, \mathbf{u}^0)$ and so are $(\nu/2)\sum_{l=1}^{L-1}(\|z_l^k - W_l^k a_{l-1}^k\|_2^2 + \|a_l^k - f(z_l^k)\|_2^2)$ and

$\|z_L^k - W_L^k a_{L-1}^k\|_2^2$. By Assumption 2, $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is bounded. By Lemma 11, it is obvious that u^k is bounded as well. □

Theorem 14. Given that Assumptions 1 and 2 hold, the dlADMM satisfies Property 2.

Proof. This follows directly from Lemma 13 and Lemma 14. □

Theorem 15. Given that Assumptions 1 and 2 hold, the dlADMM satisfies Property 3.

Proof. We know that

$$\begin{aligned}&\partial L_\rho(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1})\\ &= (\partial_{\mathbf{W}^{k+1}} L_\rho, \partial_{\mathbf{z}^{k+1}} L_\rho, \nabla_{\mathbf{a}^{k+1}} L_\rho, \nabla_{u^{k+1}} L_\rho)\end{aligned}$$

where $\partial_{\mathbf{W}^{k+1}} L_\rho = \{\partial_{W_l{}^{k+1}} L_\rho\}_{l=1}^L$, $\partial_{\mathbf{z}^{k+1}} L_\rho = \{\partial_{z_l{}^{k+1}} L_\rho\}_{l=1}^L$, and $\nabla_{\mathbf{a}^{k+1}} L_\rho = \{\nabla_{a_l{}^{k+1}} L_\rho\}_{l=1}^{L-1}$. To prove Property 3, we need to give an upper bound of $\partial_{\mathbf{W}^{k+1}} L_\rho, \partial_{\mathbf{z}^{k+1}} L_\rho, \nabla_{\mathbf{a}^{k+1}} L_\rho$ and $\nabla_{u^{k+1}} L_\rho$ by a linear combination of $\|\mathbf{W}^{k+1} - \overline{\mathbf{W}}^{k+1}\|$, $\|\mathbf{z}^{k+1} - \overline{\mathbf{z}}^{k+1}\|$, $\|\mathbf{a}^{k+1} - \overline{\mathbf{a}}^{k+1}\|$ and $\|\mathbf{z}_l^{k+1} - \mathbf{z}_l^k\|$.

For $W_l^{k+1}(l < L)$,

$$\begin{aligned}
\partial_{W_l^{k+1}} L_\rho &= \nabla_{W_l^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nu(W_l^{k+1} a_{l-1}^{k+1} - z_l^{k+1})(a_{l-1}^{k+1})^T \\
&= \nabla_{W_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\
&= \nabla_{\overline{W}_l^{k+1}} \phi(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) + \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) \\
&- \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) - \nabla_{\overline{W}_l^{k+1}} \phi(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\
&+ \nabla_{W_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\
&= \nabla_{\overline{W}_l^{k+1}} \phi(\mathbf{W}_{l-1}^{k+1}, \mathbf{z}_{l-1}^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) + \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) \\
&- \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) + \nu(W_l^{k+1} a_{l-1}^{k+1} - z_l^{k+1})(a_{l-1}^{k+1})^T \\
&- \nu(\overline{W}_l^{k+1} a_{l-1}^{k+1} - \overline{z}_l^{k+1})(a_{l-1}^{k+1})^T
\end{aligned}$$

Because

$$\begin{aligned}
&\| - \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) + \nu(W_l^{k+1} a_{l-1}^{k+1} - z_l^{k+1})(a_{l-1}^{k+1})^T - \nu(\overline{W}_l^{k+1} a_{l-1}^{k+1} \overline{z}_l^{k+1})(a_{l-1}^{k+1})^T \| \\
&= \| - \theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1}) + \nu(W_l^{k+1} - \overline{W}_l^{k+1}) a_{l-1}^{k+1} (a_{l-1}^{k+1})^T - \nu(z_l^{k+1} - \overline{z}_l^{k+1})(a_{l-1}^{k+1})^T \| \\
&\leq \|\theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1})\| + \nu \|(W_l^{k+1} - \overline{W}_l^{k+1}) a_{l-1}^{k+1} (a_{l-1}^{k+1})^T \| \\
&+ \nu \|(z_l^{k+1} - \overline{z}_l^{k+1})(a_{l-1}^{k+1})^T \| \quad \text{(Triangle Inequality)} \\
&\leq \|\theta_l^{k+1} \circ (W_l^{k+1} - \overline{W}_l^{k+1})\| + \nu \|W_l^{k+1} - \overline{W}_l^{k+1}\| \|a_{l-1}^{k+1}\| \|a_{l-1}^{k+1}\| + \nu \|z_l^{k+1} - \overline{z}_l^{k+1}\| \|a_{l-1}^{k+1}\| \\
&\text{(Cauchy-Schwarz inequality)}
\end{aligned}$$

Because a_{l-1}^{k+1} is bounded by Property 1 and Equation (A.11) holds, $\|\partial_{W_l^{k+1}} L_\rho\|$ can be upper bounded by a linear combination of $\|W_l^{k+1} - \overline{W}_l^{k+1}\|$ and $\|z_l^{k+1} - \overline{z}_l^{k+1}\|$.

For W_L^{k+1},

$$\begin{aligned}
\partial_{W_L^{k+1}} L_\rho &= \nabla_{W_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nabla_{\overline{W}_L^{k+1}} \phi(\mathbf{W}_{L-1}^{k+1}, \mathbf{z}_{L-1}^{k+1}, \mathbf{a}^{k+1}, u^k) + \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) \\
&- \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) - \nabla_{\overline{W}_L^{k+1}} \phi(\mathbf{W}_{L-1}^{k+1}, \mathbf{z}_{L-1}^{k+1}, \mathbf{a}^{k+1}, u^k) \\
&+ \nabla_{W_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nabla_{W_L^{k+1}} \phi(\mathbf{W}_{L-1}^{k+1}, \mathbf{z}_{L-1}^{k+1}, \mathbf{a}^{k+1}, u^k) + \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) \\
&- \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) + \rho(W_L^{k+1} a_{L-1}^{k+1} - z_L^{k+1} - u^{k+1}/\rho)(a_{L-1}^{k+1})^T \\
&- \rho(\overline{W}_L^{k+1} a_{L-1}^{k+1} - \overline{z}_L^{k+1} - u^k/\rho)(a_{L-1}^{k+1})^T
\end{aligned}$$

Because

$$\begin{aligned}
&\| - \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) + \rho(W_L^{k+1} a_{L-1}^{k+1} - z_L^{k+1} - u^{k+1}/\rho)(a_{L-1}^{k+1})^T \\
&- \rho(\overline{W}_L^{k+1} a_{L-1}^{k+1} - \overline{z}_L^{k+1} - u^k/\rho)(a_{L-1}^{k+1})^T \| \\
&= \| - \theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1}) + \rho(W_L^{k+1} - \overline{W}_L^{k+1}) a_{L-1}^{k+1} (a_{L-1}^{k+1})^T - \rho(z_L^{k+1} - \overline{z}_L^{k+1})(a_{L-1}^{k+1})^T \\
&- (u^{k+1} - u^k)(a_{L-1}^{k+1})^T \| \\
&\leq \|\theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1})\| + \rho \|(W_L^{k+1} - \overline{W}_L^{k+1}) a_{L-1}^{k+1} (a_{L-1}^{k+1})^T\| \\
&+ \rho \|(z_L^{k+1} - \overline{z}_L^{k+1})(a_{L-1}^{k+1})^T\| + \|(u^{k+1} - u^k)(a_{L-1}^{k+1})^T\| \quad \text{(Triangle Inequality)} \\
&\leq \|\theta_L^{k+1} \circ (W_L^{k+1} - \overline{W}_L^{k+1})\| + \rho \|W_L^{k+1} - \overline{W}_L^{k+1}\| \|a_{L-1}^{k+1}\| \|a_{L-1}^{k+1}\| + \rho \|z_L^{k+1} - \overline{z}_L^{k+1}\| \|a_{L-1}^{k+1}\| \\
&+ H \|z_L^{k+1} - z_L^k\| \|a_{L-1}^{k+1}\|
\end{aligned}$$

(Cauchy-Schwarz inequality, Lemma 11, $R(z_L; y)$ is Lipschitz differentiable)

Because a_{L-1}^{k+1} is bounded by Property 1 and Equation (A.11) holds, $\|\partial_{W_L^{k+1}} L_\rho\|$ can be upper bounded by a linear combination of $\|W_L^{k+1} - \overline{W}_L^{k+1}\|$, $\|z_L^{k+1} - \overline{z}_L^{k+1}\|$ and $\|z_L^{k+1} - z_L^k\|$.

For $z_l^{k+1}(l < L)$,

$$\begin{aligned}
\partial_{z_l^{k+1}} L_\rho &= \partial_{z_l^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nu(z_l^{k+1} - W_l^{k+1} a_{l-1}^{k+1}) + \nu \partial f_l(z_l^{k+1}) \circ (f(z_l^{k+1}) - a_l^{k+1}) \\
&= \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_l^{k+1}, u^k) \\
&= \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_l^{k+1}, u^k) - \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\
&+ \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k) \\
&= \nu \partial f_l(z_l^{k+1}) \circ (\overline{a}_l^{k+1} - a_l^{k+1}) + \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k)
\end{aligned}$$

because z_l^{k+1} is bounded and $f_l(z_l)$ is continuous and hence $f_l(z_l^{k+1})$ is bounded, and the optimality condition of Equation (2.10) yields

$$0 \in \partial_{z_l^{k+1}} \phi(\mathbf{W}_l^{k+1}, \mathbf{z}_l^{k+1}, \mathbf{a}_{l-1}^{k+1}, u^k)$$

Therefore, $\|\partial_{z_l^{k+1}} L_\rho\|$ is upper bounded by $\|a_l^{k+1} - \overline{a}_l^{k+1}\|$.

For z_L^{k+1},

$$\begin{aligned}
\nabla_{z_L^{k+1}} L_\rho &= \nabla_{z_L^{k+1}} R(z_L^{k+1}; y) + \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nabla_{z_L^{k+1}} R(z_L^{k+1}; y) + \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^k) \\
&- \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^k) + \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\
&= \nabla_{z_L^{k+1}} R(z_L^{k+1}; y) + \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^k) + u^{k+1} - u^k
\end{aligned}$$

The optimality condition of Equation (2.11) yields

$$0 \in \nabla_{z_L^{k+1}} R(z_L^{k+1}; y) + \nabla_{z_L^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^k)$$

and

$$\|u^{k+1}-u^k\| = \|\nabla_{z_L^{k+1}} R(z_L^{k+1};y) - \nabla_{z_L^k} R(z_L^k;y)\| \text{ (Lemma 11)} \leq H\|z_L^{k+1}-z_L^k\|$$

(Cauchy-Schwarz inequality, $R(z_L;y)$ is Lipschitz differentiable)

Therefore $\|\nabla_{z_L^{k+1}} L_\rho\|$ is upper bounded by $\|z_L^{k+1}-z_L^k\|$.

For $a_l^{k+1}(l<L-1)$,

$$\begin{aligned}
&\nabla_{a_l^{k+1}}\phi(\mathbf{W}^{k+1},\mathbf{z}^{k+1},\mathbf{a}^{k+1},u^{k+1})\\
&=\nu(W_{l+1}^{k+1})^T(W_{l+1}^{k+1}a_l^{k+1}-z_{l+1}^{k+1})+\nu(a_l^{k+1}-f_l(z_l^{k+1}))\\
&=\nabla_{a_l^{k+1}}\phi(\mathbf{W}_{l+1}^{k+1},\mathbf{z}_{l+1}^{k+1},\mathbf{a}_l^{k+1},u^k)\\
&=\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})+\nabla_{\overline{a}_l^{k+1}}\phi(\mathbf{W}_l^{k+1},\mathbf{z}_l^{k+1},\mathbf{a}_{l-1}^{k+1},u^k)-\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})\\
&-\nabla_{\overline{a}_l^{k+1}}\phi(\mathbf{W}_l^{k+1},\mathbf{z}_l^{k+1},\mathbf{a}_{l-1}^{k+1},u^k)+\nabla_{a_l^{k+1}}\phi(\mathbf{W}_{l+1}^{k+1},\mathbf{z}_{l+1}^{k+1},\mathbf{a}_l^{k+1},u^k)\\
&=\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})+\nabla_{\overline{a}_l^{k+1}}\phi(\mathbf{W}_l^{k+1},\mathbf{z}_l^{k+1},\mathbf{a}_{l-1}^{k+1},u^k)-\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})\\
&-\nu(\overline{W}_{l+1}^{k+1})^T(\overline{W}_{l+1}^{k+1}\overline{a}_l^{k+1}-\overline{z}_{l+1}^{k+1})-\nu(\overline{a}_l^{k+1}-f_l(z_l^{k+1}))\\
&+\nu(W_{l+1}^{k+1})^T(W_{l+1}^{k+1}a_l^{k+1}-z_{l+1}^{k+1})+\nu(a_l^{k+1}-f_l(z_l^{k+1}))
\end{aligned}$$

Because

$$\begin{aligned}
&\|-\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})-\nu(\overline{W}_{l+1}^{k+1})^T(\overline{W}_{l+1}^{k+1}\overline{a}_l^{k+1}-\overline{z}_{l+1}^{k+1})-\nu(\overline{a}_l^{k+1}-f_l(z_l^{k+1}))\\
&+\nu(W_{l+1}^{k+1})^T(W_{l+1}^{k+1}a_l^{k+1}-z_{l+1}^{k+1})+\nu(a_l^{k+1}-f_l(z_l^{k+1}))\|\\
&\leq\|\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})\|+\nu\|a_l^{k+1}-\overline{a}_l^{k+1}\|+\nu\|(W_{l+1}^{k+1})^Tz_{l+1}^{k+1}-(\overline{W}_{l+1}^{k+1})^T\overline{z}_{l+1}^{k+1}\|\\
&+\nu\|(W_{l+1}^{k+1})^TW_{l+1}^{k+1}a_l^{k+1}-(\overline{W}_{l+1}^{k+1})^T\overline{W}_{l+1}^{k+1}\overline{a}_l^{k+1}\|
\end{aligned}$$

(Triangle Inequality)

Then we need to show that $\nu\|(W_{l+1}^{k+1})^Tz_{l+1}^{k+1}-(\overline{W}_{l+1}^{k+1})^T\overline{z}_{l+1}^{k+1}\|$ and $\nu\|(W_{l+1}^{k+1})^TW_{l+1}^{k+1}a_l^{k+1}-(\overline{W}_{l+1}^{k+1})^T\overline{W}_{l+1}^{k+1}\overline{a}_l^{k+1}\|$ are upper bounded by $\|\mathbf{W}^{k+1}-\overline{\mathbf{W}}^{k+1}\|$, $\|\mathbf{z}^{k+1}-\overline{\mathbf{z}}^{k+1}\|$, and

$\|\mathbf{a}^{k+1} - \overline{\mathbf{a}}^{k+1}\|$.

$$
\begin{aligned}
&\nu\|(W_{l+1}^{k+1})^T z_{l+1}^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{z}_{l+1}^{k+1}\| \\
&= \nu\|(W_{l+1}^{k+1})^T z_{l+1}^{k+1} - (\overline{W}_{l+1}^{k+1})^T z_{l+1}^{k+1} + (\overline{W}_{l+1}^{k+1})^T z_{l+1}^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{z}_{l+1}^{k+1}\| \\
&\leq \nu\|z_{l+1}^{k+1}\|\|W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}\| + \nu\|\overline{W}_{l+1}^{k+1}\|\|z_{l+1}^{k+1} - \overline{z}_{l+1}^{k+1}\| \\
&\text{(Triangle Inequality, Cauchy-Schwarz inequality)}
\end{aligned}
$$

Because $\|z_{l+1}^{k+1}\|$ and $\|\overline{W}_{l+1}^{k+1}\|$ are upper bounded, $\nu\|(W_{l+1}^{k+1})^T z_{l+1}^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{z}_{l+1}^{k+1}\|$ is therefore upper bounded by a combination of $\|W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}\|$ and $\|z_{l+1}^{k+1} - \overline{z}_{l+1}^{k+1}\|$.

$$
\begin{aligned}
&\nu\|(W_{l+1}^{k+1})^T W_{l+1}^{k+1} a_l^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1}\| \\
&= \nu\|(W_{l+1}^{k+1})^T W_{l+1}^{k+1} a_l^{k+1} - (W_{l+1}^{k+1})^T W_{l+1}^{k+1} \overline{a}_l^{k+1} + (W_{l+1}^{k+1})^T W_{l+1}^{k+1} \overline{a}_l^{k+1} \\
&- (W_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1} + (W_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1}\| \\
&\leq \nu\|(W_{l+1}^{k+1})^T W_{l+1}^{k+1} (a_l^{k+1} - \overline{a}_l^{k+1})\| + \nu\|(W_{l+1}^{k+1})^T (W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}) \overline{a}_l^{k+1}\| \\
&+ \nu\|(W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1}\| \text{(Triangle Inequality)} \\
&\leq \nu\|W_{l+1}^{k+1}\|\|W_{l+1}^{k+1}\|\|a_l^{k+1} - \overline{a}_l^{k+1}\| + \nu\|W_{l+1}^{k+1}\|\|W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}\|\|\overline{a}_l^{k+1}\| \\
&+ \nu\|W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}\|\|\overline{W}_{l+1}^{k+1}\|\|\overline{a}_l^{k+1}\| \text{(Cauchy-Schwarz Inequality)}
\end{aligned}
$$

Because $\|W_{l+1}^{k+1}\|$, $\|\overline{W}_{l+1}^{k+1}\|$ and $\|\overline{a}_l^{k+1}\|$ are upper bounded, $\nu\|(W_{l+1}^{k+1})^T W_{l+1}^{k+1} a_l^{k+1} - (\overline{W}_{l+1}^{k+1})^T \overline{W}_{l+1}^{k+1} \overline{a}_l^{k+1}\|$ is therefore upper bounded by a combination of $\|W_{l+1}^{k+1} - \overline{W}_{l+1}^{k+1}\|$ and $\|a_{l+1}^{k+1} - \overline{a}_{l+1}^{k+1}\|$.

For a_{L-1}^{k+1},

$$
\begin{aligned}
&\nabla_{a_{L-1}^{k+1}}\phi(\mathbf{W}^{k+1},\mathbf{z}^{k+1},\mathbf{a}^{k+1},u^{k+1})\\
&=\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})+\nabla_{\overline{a}_{L-1}^{k+1}}\phi(\mathbf{W}_{L-1}^{k+1},\mathbf{z}_{L-1}^{k+1},\mathbf{a}_{L-2}^{k+1},u^{k})-\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})\\
&-\nabla_{\overline{a}_{L-1}^{k+1}}\phi(\mathbf{W}_{L-1}^{k+1},\mathbf{z}_{L-1}^{k+1},\mathbf{a}_{L-2}^{k+1},u^{k})+\nabla_{a_{L-1}^{k+1}}\phi(\mathbf{W}^{k+1},\mathbf{z}^{k+1},\mathbf{a}^{k+1},u^{k+1})\\
&=\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})+\nabla_{\overline{a}_{L-1}^{k+1}}\phi(\mathbf{W}_{L-1}^{k+1},\mathbf{z}_{L-1}^{k+1},\mathbf{a}_{L-2}^{k+1},u^{k})-\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})\\
&-\rho(\overline{W}_L^{k+1})^T(\overline{W}_L^{k+1}\overline{a}_{L-1}^{k+1}-\overline{z}_L^{k+1}-u^k/\rho)-\nu(\overline{a}_{L-1}^{k+1}-f_{L-1}(z_{L-1}^{k+1}))\\
&+\rho(W_L^{k+1})^T(W_L^{k+1}a_{L-1}^{k+1}-z_L^{k+1}-u^{k+1}/\rho)+\nu(a_{L-1}^{k+1}-f_{L-1}(z_{L-1}^{k+1}))
\end{aligned}
$$

Because

$$
\begin{aligned}
&\|-\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})-\rho(\overline{W}_L^{k+1})^T(\overline{W}_L^{k+1}\overline{a}_{L-1}^{k+1}-\overline{z}_L^{k+1}-u^k/\rho)\\
&-\nu(\overline{a}_{L-1}^{k+1}-f_{L-1}(z_{L-1}^{k+1}))+\rho(W_L^{k+1})^T(W_L^{k+1}a_{L-1}^{k+1}-z_L^{k+1}-u^{k+1}/\rho)\\
&+\nu(a_{L-1}^{k+1}-f_{L-1}(z_{L-1}^{k+1}))\|\\
&\leq\|\tau_{L-1}^{k+1}\circ(a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1})\|+\nu\|a_{L-1}^{k+1}-\overline{a}_{L-1}^{k+1}\|\\
&+\rho\|(W_L^{k+1})^Tz_L^{k+1}-(\overline{W}_L^{k+1})^T\overline{z}_L^{k+1}\|+\|(W_L^{k+1})^Tu^{k+1}-(\overline{W}_L^{k+1})^Tu^k\|\\
&+\rho\|(W_L^{k+1})^TW_L^{k+1}a_{L-1}^{k+1}-(\overline{W}_L^{k+1})^T\overline{W}_L^{k+1}\overline{a}_{L-1}^{k+1}\|(\text{Triangle Inequality})
\end{aligned}
$$

Then we need to show that $\rho\|(W_L^{k+1})^Tz_L^{k+1}-(\overline{W}_L^{k+1})^T\overline{z}_L^{k+1}\|$, $\|(W_L^{k+1})^Tu^{k+1}-(\overline{W}_L^{k+1})^Tu^k\|$ and $\rho\|(W_L^{k+1})^TW_L^{k+1}a_{L-1}^{k+1}-(\overline{W}_L^{k+1})^T\overline{W}_L^{k+1}\overline{a}_{L-1}^{k+1}\|$ are upper bounded by $\|\mathbf{W}^{k+1}-\overline{\mathbf{W}}^{k+1}\|$, $\|\mathbf{z}^{k+1}-\overline{\mathbf{z}}^{k+1}\|$, $\|\mathbf{a}^{k+1}-\overline{\mathbf{a}}^{k+1}\|$ and $\|\mathbf{z}^{k+1}-\mathbf{z}^k\|$.

$$
\begin{aligned}
&\rho\|(W_L^{k+1})^Tz_L^{k+1}-(\overline{W}_L^{k+1})^T\overline{z}_L^{k+1}\|\\
&=\rho\|(W_L^{k+1})^Tz_L^{k+1}-(\overline{W}_L^{k+1})^Tz_L^{k+1}+(\overline{W}_L^{k+1})^Tz_L^{k+1}-(\overline{W}_L^{k+1})^T\overline{z}_L^{k+1}\|\\
&\leq\rho\|z_L^{k+1}\|\|W_L^{k+1}-\overline{W}_L^{k+1}\|+\rho\|\overline{W}_L^{k+1}\|\|z_L^{k+1}-\overline{z}_L^{k+1}\|\\
&(\text{Triangle Inequality, Cauchy-Schwarz inequality})
\end{aligned}
$$

Because $\|z_L^{k+1}\|$ and $\|\overline{W}_L^{k+1}\|$ are upper bounded, $\rho\|(W_L^{k+1})^T z_L^{k+1} - (\overline{W}_L^{k+1})^T \overline{z}_L^{k+1}\|$ is therefore upper bounded by a combination of $\|W_L^{k+1} - \overline{W}_L^{k+1}\|$ and $\|z_L^{k+1} - \overline{z}_L^{k+1}\|$.

$$
\begin{aligned}
&\|(W_L^{k+1})^T u^{k+1} - (\overline{W}_L^{k+1})^T u^k\| \\
&= \|(W_L^{k+1})^T u^{k+1} - (W_L^{k+1})^T u^k + (W_L^{k+1})^T u^k - (\overline{W}_L^{k+1})^T u^k\| \\
&\leq \|(W_L^{k+1})^T (u^{k+1} - u^k)\| + \|(W_L^{k+1} - \overline{W}_L^{k+1})^T u^k\| \text{(Triangle Inequality)} \\
&\leq \|W_L^{k+1}\|\|u^{k+1} - u^k\| + \|W_L^{k+1} - \overline{W}_L^{k+1}\|\|u^k\| \text{(Cauthy-Schwarz inequality)} \\
&= \|W_L^{k+1}\|\|\nabla_{z_L^{k+1}} R(z_L^{k+1}; y) - \nabla_{z_L^k} R(z_L^k; y)\| + \|W_L^{k+1} - \overline{W}_L^{k+1}\|\|u^k\| \text{ (Lemma 11)} \\
&\leq H\|W_L^{k+1}\|\|z_L^{k+1} - z_L^k\| + \|W_L^{k+1} - \overline{W}_L^{k+1}\|\|u^k\| \\
&(R(z_L; y) \text{ is Lipschitz differentiable})
\end{aligned}
$$

Because $\|W_L^{k+1}\|$ and $\|u^k\|$ are bounded, $\|(W_L^{k+1})^T u^{k+1} - (\overline{W}_L^{k+1})^T u^k\|$ can be upper bounded by a combination of $\|z_L^{k+1} - z_L^k\|$ and $\|W_L^{k+1} - \overline{W}_L^{k+1}\|$.

$$
\begin{aligned}
&\rho\|(W_L^{k+1})^T W_L^{k+1} a_{L-1}^{k+1} - (\overline{W}_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1}\| \\
&= \rho\|(W_L^{k+1})^T W_L^{k+1} a_{L-1}^{k+1} - (W_L^{k+1})^T W_L^{k+1} \overline{a}_{L-1}^{k+1} + (W_L^{k+1})^T W_L^{k+1} \overline{a}_{L-1}^{k+1} \\
&- (W_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1} + (W_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1} - (\overline{W}_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1}\| \\
&\leq \rho\|(W_L^{k+1})^T W_L^{k+1} (a_{L-1}^{k+1} - \overline{a}_{L-1}^{k+1})\| + \rho\|(W_L^{k+1})^T (W_L^{k+1} - \overline{W}_L^{k+1}) \overline{a}_{L-1}^{k+1}\| \\
&+ \rho\|(W_L^{k+1} - \overline{W}_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1}\| \text{(Triangle Inequality)} \\
&\leq \rho\|W_L^{k+1}\|\|W_L^{k+1}\|\|a_{L-1}^{k+1} - \overline{a}_{L-1}^{k+1}\| + \rho\|W_L^{k+1}\|\|W_L^{k+1} - \overline{W}_L^{k+1}\|\|\overline{a}_{L-1}^{k+1}\| \\
&+ \rho\|W_L^{k+1} - \overline{W}_L^{k+1}\|\|\overline{W}_L^{k+1}\|\|\overline{a}_{L-1}^{k+1}\| \text{(Cauchy-Schwarz inequality)}
\end{aligned}
$$

Because $\|W_L^{k+1}\|$, $\|\overline{W}_L^{k+1}\|$ and $\|\overline{a}_{L-1}^{k+1}\|$ are upper bounded, $\rho\|(W_L^{k+1})^T W_L^{k+1} a_{L-1}^{k+1} - (\overline{W}_L^{k+1})^T \overline{W}_L^{k+1} \overline{a}_{L-1}^{k+1}\|$ is therefore upper bounded by a combination of $\|W_L^{k+1} - \overline{W}_L^{k+1}\|$ and $\|a_L^{k+1} - \overline{a}_L^{k+1}\|$.

For u^{k+1},

$$\begin{aligned}\nabla_{u_l^{k+1}} L_\rho &= \nabla_{u_l^{k+1}} \phi(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}, u^{k+1}) \\ &= z_L^{k+1} - W_L^{k+1} a_L^{k+1} \\ &= (1/\rho)(u^{k+1} - u^k) \\ &= (1/\rho)(\nabla_{z_L^k} R(z_L^k; y) - \nabla_{z_L^{k+1}} R(z_L^{k+1}; y))(\text{Lemma 11})\end{aligned}$$

Because

$$\begin{aligned}&\|(1/\rho)(\nabla_{z_L^k} R(z_L^k; y) - \nabla_{z_L^{k+1}} R(z_L^{k+1}; y))\| \\ &\leq (H/\rho)\|z_L^{k+1} - z_L^k\|(R(z_L; y) \text{ is Lipschitz differentiable})\end{aligned}$$

Therefore, $\|\nabla_{u_l^{k+1}} L_\rho\|$ is upper bounded by $\|z_L^{k+1} - z_L^k\|$. □

A.4 Proof of Theorem 3

Proof. To prove this theorem, we will first show that c_k satisfies two conditions: (1). $c_k \geq c_{k+1}$. (2). $\sum_{k=0}^{\infty} c_k$ is bounded. We then conclude the convergence rate of $o(1/k)$ based on these two conditions. Specifically, first, we have

$$\begin{aligned}c_k &= \min_{0\leq i\leq k}(\sum_{l=1}^{L}(\|\overline{\theta}_l^{i+1} \circ (\overline{W}_l^{i+1} - W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{i+1} \circ (\overline{W}_l^{i+1} - \overline{W}_l^{i+1})^{\circ 2}\|_1/2) \\ &\quad + \|\overline{\tau}_l^{i+1} \circ (\overline{a}_l^{i+1} - a_l^i)^{\circ 2}\|_1/2 + \|\tau_l^{i+1} \circ (a_l^{i+1} - \overline{a}_l^{i+1})^{\circ 2}\|_1/2) + C_2\|\overline{z}_L^{i+1} - z_L^i\|_2^2 \\ &\quad + C_1\|z_L^{i+1} - \overline{z}_L^{i+1}\|_2^2) \\ &\quad \geq \min_{0\leq i\leq k+1}(\sum_{l=1}^{L}(\|\overline{\theta}_l^{i+1} \circ (\overline{W}_l^{i+1} - W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{i+1} \circ (\overline{W}_l^{i+1} - \overline{W}_l^{i+1})^{\circ 2}\|_1/2) \\ &\quad + \sum_{l=1}^{L-1}(\|\overline{\tau}_l^{i+1} \circ (\overline{a}_l^{i+1} - a_l^i)^{\circ 2}\|_1/2 + \|\tau_l^{i+1} \circ (a_l^{i+1} - \overline{a}_l^{i+1})^{\circ 2}\|_1/2) + C_2\|\overline{z}_L^{i+1} - z_L^i\|_2^2 \\ &\quad + C_1\|z_L^{i+1} - \overline{z}_L^{i+1}\|_2^2) \\ &\quad = c_{k+1}\end{aligned}$$

Therefore c_k satisfies the first condition. Second,

$$
\begin{aligned}
&\sum_{k=0}^{\infty} c_k \\
&= \sum_{k=0}^{\infty} \min_{0\le i\le k}\Big(\sum_{l=1}^{L}(\|\overline{\theta}_l^{i+1}\circ(\overline{W}_l^{i+1}-W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{i+1}\circ(\overline{W}_l^{i+1}-\overline{W}_l^{i+1})^{\circ 2}\|_1/2) \\
&+ \sum_{l=1}^{L-1}(\|\overline{\tau}_l^{i+1}\circ(\overline{a}_l^{i+1}-a_l^i)^{\circ 2}\|_1/2 + \|\tau_l^{i+1}\circ(a_l^{i+1}-\overline{a}_l^{i+1})^{\circ 2}\|_1/2) + C_2\|\overline{z}_L^{i+1}-z_L^i\|_2^2 \\
&+ C_1\|z_L^{i+1}-\overline{z}_L^{i+1}\|_2^2\Big) \\
&\le \sum_{k=0}^{\infty}\Big(\sum_{l=1}^{L}(\|\overline{\theta}_l^{k+1}\circ(\overline{W}_l^{k+1}-W_l^k)^{\circ 2}\|_1/2 + \|\theta_l^{k+1}\circ(\overline{W}_l^{k+1}-\overline{W}_l^{k+1})^{\circ 2}\|_1/2) \\
&+ \sum_{l=1}^{L-1}(\|\overline{\tau}_l^{k+1}\circ(\overline{a}_l^{k+1}-a_l^k)^{\circ 2}\|_1/2 + \|\tau_l^{k+1}\circ(a_l^{k+1}-\overline{a}_l^{k+1})^{\circ 2}\|_1/2) + C_2\|\overline{z}_L^{k+1}-z_L^k\|_2^2 \\
&+ C_1\|z_L^{k+1}-\overline{z}_L^{k+1}\|_2^2\Big) \\
&\le L_\rho(\mathbf{W}^0,\mathbf{z}^0,\mathbf{a}^0,u^0) - L_\rho(\mathbf{W}^*,\mathbf{z}^*,\mathbf{a}^*,u^*) \quad \text{(Property 2)}
\end{aligned}
$$

So $\sum_{k=0}^{\infty} c_k$ is bounded and c_k satisfies the second condition. Finally, it has been proved that the sufficient conditions of convergence rate $o(1/k)$ are: (1) $c_k \ge c_{k+1}$, and (2) $\sum_{k=0}^{\infty} c_k$ is bounded, and (3) $c_k \ge 0$ (Lemma 1.2 in [26]). Since we have proved the first two conditions and the third one $c_k \ge 0$ is obvious, the convergence rate of $o(1/k)$ is proven. □

Appendix B

Appendix of the mDLAM Algorithm

B.1 Definition

Several definitions are shown here for the sake of convergence analysis.

Definition 7 (Coercivity)**.** Any arbitrary function $G_2(x)$ is coercive over a nonempty set $dom(G_2)$ if as $\|x\| \to \infty$ and $x \in dom(G_2)$, we have $G_2(x) \to \infty$, where $dom(G_2)$ is a domain set of G_2.

Definition 8 (Multi-convexity)**.** A function $f(x_1, x_2, \cdots, x_m)$ is a multi-convex function if f is convex with regard to $x_i (i = 1, \cdots, m)$ while fixing other variables.

Definition 9 (Lipschitz Differentiability)**.** A function $f(x)$ is Lipschitz differentiable with Lipschitz coefficient $L > 0$ if for any $x_1, x_2 \in \mathbb{R}$, the following inequality holds:

$$\|\nabla f(x_1) - \nabla f(x_2)\| \leq L\|x_1 - x_2\|.$$

For Lipschitz differentiability, we have the following lemma (Lemma 2.1 in [2]):

Lemma 15. If $f(x)$ is Lipschitz differentiable with $L > 0$, then for any $x_1, x_2 \in \mathbb{R}$

$$f(x_1) \leq f(x_2) + \nabla f^T(x_2)(x_1 - x_2) + \frac{L}{2}\|x_1 - x_2\|^2.$$

Definition 10 (Fréchet Subdifferential)**.** For each $x_1 \in dom(u_1)$, the Fréchet subdifferential of u_1 at x_1, which is denoted as $\hat{\partial} u_1(x_1)$, is the set of vectors v, which satisfy

$$\lim_{x_2 \neq x_1} \inf_{x_2 \to x_1} (u_1(x_2) - u_1(x_1) - v^T(x_2 - x_1))/\|x_2 - x_1\| \geq 0.$$

The vector $v \in \hat{\partial} u_1(x_1)$ is a Fréchet subgradient.

Then the definition of the limiting subdifferential, which is based on Fréchet subdifferential, is given in the following [84]:

Definition 11 (Limiting Subdifferential)**.** For each $x \in dom(u_2)$, the limiting subdifferential (or subdifferential) of u_2 at x is

$$\partial u_2(x) = \{v_1 | \exists\ x^k \to x, s.t.\ u_2(x^k) \to u_2(x), v^k \in \hat{\partial} u_2(x^k), v^k \to v\}.$$

where x^k is a sequence whose limit is x and the limit of $u_2(x^k)$ is $u_2(x)$, v^k is a sequence, which is a Fréchet subgradient of u_2 at x^k and whose limit is v. The vector $v \in \partial u_2(x)$ is a limiting subgradient.

Specifically, when u_2 is convex, its limiting subdifferential is reduced to regular subdifferential [84], which is defined as follows:

Definition 12 (Regular Subdifferential)**.** For each $x_1 \in dom(f)$, the regular subdifferential of a convex function f at x_1, which is denoted as $\partial f(x_1)$, is the set of vectors

v, which satisfy

$$f(x_2) \geq f(x_1) + v^T(x_2 - x_1).$$

The vector $v \in \partial f(x_1)$ is a regular subgradient.

Definition 13 (Quasilinearity)**.** A function $f(x)$ is quasiconvex if for any sublevel set $S_\nu(f) = \{x|f(x) \leq \nu\}$ is a convex set. Likewise, A function $f(x)$ is quasiconcave if for any superlevel set $S_\nu(f) = \{x|f(x) \geq \nu\}$ is a convex set. A function $f(x)$ is quasilinear if it is both quasiconvex and quasiconcave.

Definition 14 (Locally Strong Convexity)**.** A function $f(x)$ is locally strongly convex within a bound set $\mathbb{D}$ with a constant μ if

$$f(y) \geq f(x) + g^T(y - x) + \frac{\mu}{2}\|x - y\|_2^2 \ \forall \ g \in \partial f(x) \text{ and } \ x, y \in \mathbb{D}.$$

Simply speaking, a locally strongly convex function lies above a quadratic function within a bounded set.

Definition 15 (Kurdyka-Lojasiewicz (KL) Property)**.** A function $f(x)$ has the KL Property at $\overline{x} \in dom\ \partial f = \{x \in \mathbb{R} : \partial f(x) \neq \emptyset\}$ if there exists $\eta \in (0, +\infty]$, a neighborhood X of $\overline{x}$ and a function $\psi \in \Psi_\eta$, such that for all

$$x \in X \cap \{x \in \mathbb{R} : f(\overline{x}) < f(x) < f(\overline{x}) + \eta\},$$

the following inequality holds

$$\psi^{'}(f(x) - f(\overline{x}))dist(0, \partial f(x)) \geq 1,$$

where Ψ_η stands for a class of function $\psi : [0, \eta] \to \mathbb{R}^+$ satisfying: (1). ϕ is concave and $\psi^{'}(x)$ continuous on $(0, \eta)$; (2). ψ is continuous at 0, $\psi(0) = 0$; and (3). $\psi^{'}(x) >$

$0, \forall x \in (0, \eta)$.

The following lemma shows that a locally strongly convex function satisfies the KL Property:

Lemma 16 ([115])**.** A locally strongly convex function $f(x)$ with a constant μ satisfies the KL Property at any $x \in \mathbb{D}$ with $\psi(x) = \frac{2}{\mu}\sqrt{x}$ and $X = \mathbb{D} \cap \{y : f(y) \geq f(x)\}$.

B.2 Preliminary Lemmas

In this section, we present preliminary lemmas of the proposed mDLAM algorithm. The limiting subdifferential is used to prove the convergence of the proposed mDLAM algorithm in the following convergence analysis. Without loss of generality, ∂R is assumed to be nonempty, and the limiting subdifferential of F defined in Problem 3 is [115]:

$$\partial F(\mathbf{W}, \mathbf{z}, \mathbf{a}) = \partial_{\mathrm{W}} F \times \partial_{\mathrm{z}} F \times \partial_{\mathrm{a}} F,$$

where $\times$ means the Cartesian product.

Lemma 17. If Equation (3.3) holds, then

$$\nabla_{\overline{W}_l^{k+1}} \phi + \theta_l^{k+1}(W_l^{k+1} - \overline{W}_l^{k+1}) = 0.$$

Likewise, if Equation (3.4) holds, then there exists q such that

$$\nabla_{\overline{z}_l^{k+1}} \phi + \rho(z_l^{k+1} - \overline{z}_l^{k+1}) + q = 0,$$

where q is a subgradient with regard to z_l^{k+1} to satisfy the constraint $f_l(z_l^{k+1}) - \varepsilon \leq a_l^k \leq f_l(z_l^{k+1}) + \varepsilon$.

If Equation (3.5) holds, then there exists $u \in \partial R(z_L^{k+1}; y)$ such that

$$\nabla_{\overline{z}_L^{k+1}} \phi + \rho(z_L^{k+1} - \overline{z}_L^{k+1}) + u = 0.$$

If Equation (3.6) holds, then there exists v such that

$$\nabla_{\overline{a}_l^{k+1}} \phi + \tau_l^{k+1}(a_l^{k+1} - \overline{a}_l^{k+1}) + v = 0,$$

where v is a subgradient with regard to a_l^{k+1} to satisfy the constraint $f_l(z_l^{k+1}) - \varepsilon \leq a_l^{k+1} \leq f_l(z_l^{k+1}) + \varepsilon$.

Proof. These can be obtained by directly applying the optimality conditions of Equation (3.3), Equation (3.4), Equation (3.5) and Equation (3.6), respectively. □

Lemma 18. For Equation (3.4) and Equation (3.5), the following inequalities hold:

$$V_l^{k+1}(z_l^{k+1}) \geq \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^{k+1}). \tag{B.1}$$

Proof. Because $\phi(a_{l-1}, W_l, z_l)$ is Lipschitz differentiable with respect to z_l with Lipschitz coefficient ρ, we directly apply Lemma 15 to ϕ to obtain Equation (B.1). □

B.3 Main Proofs

Proof of Lemma 1

Proof. In Algorithm 4, we only show Equation (3.7) because Equation (3.8) and Equation (3.9) follow the same routine of Equation (3.7).

In Line 7 of Algorithm 4, if $F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) < F(\mathbf{W}_{\leq l-1}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$, then obviously there exists $\alpha_l^{k+1} > 0$ such that Equation (3.7) holds. Otherwise, according

to Line 8 of Algorithm 4, because $\phi(a_{l-1}, W_l, z_l)$ is convex with regard to W_l, according to the definition of regular subgradient, we have

$$\phi(a_{l-1}^{k+1}, W_l^k, z_l^k) \geq \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + \nabla_{\overline{W}_l^{k+1}} \phi^T (W_l^k - \overline{W}_l^{k+1}), \tag{B.2}$$

Therefore, we have

$$\begin{aligned}
&F(\mathbf{W}_{\leq l-1}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) - F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \\
&= \phi(a_{l-1}^{k+1}, W_l^k, z_l^k) - \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^k) \quad \text{(Definition of } F \text{ in Problem 3)} \\
&\geq -(\nabla_{\overline{W}_l^{k+1}} \phi)^T (W_l^{k+1} - \overline{W}_l^{k+1}) - \frac{\theta_l^{k+1}}{2} \|W_l^{k+1} - \overline{W}_l^{k+1}\|_2^2 \\
&- \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + \phi(a_{l-1}^{k+1}, W_l^k, z_l^k) \quad \text{(Equation (3.2))} \\
&\geq -(\nabla_{\overline{W}_l^{k+1}} \phi)^T (W_l^{k+1} - W_l^k) - \frac{\theta_l^{k+1}}{2} \|W_l^{k+1} - \overline{W}_l^{k+1}\|_2^2 \quad \text{(Equation (B.2))} \\
&= \theta_l^{k+1} (W_l^{k+1} - \overline{W}_l^{k+1})^T (W_l^{k+1} - W_l^k) - \frac{\theta_l^{k+1}}{2} \|W_l^{k+1} - \overline{W}_l^{k+1}\|_2^2 \quad \text{(Lemma 17)} \\
&= \frac{\theta_l^{k+1}}{2} \|W_l^{k+1} - W_l^k\|_2^2 \quad (\overline{W}_l^{k+1} = W_l^k).
\end{aligned}$$

Let $\alpha_l^{k+1} = \theta_l^{k+1}$, then Equation (3.7) still holds. □

Proof of Lemma 3

Proof. In Algorithm 4:

(a). We sum Equation (3.7), Equation (3.8) and Equation (3.9) from $l = 1$ to L and from $k = 0$ to K to obtain

$$\begin{aligned}
&F(\mathbf{W}^0, \mathbf{z}^0, \mathbf{a}^0) - F(\mathbf{W}^K, \mathbf{z}^K, \mathbf{a}^K) \\
&\geq \sum_{k=0}^{K} (\sum_{l=1}^{L} (\frac{\alpha_l^{k+1}}{2} \|W_l^{k+1} - W_l^k\|_2^2 + \frac{\gamma_l^{k+1}}{2} \|z_l^{k+1} - z_l^k\|_2^2) \\
&+ \sum_{l=1}^{L-1} \frac{\delta_l^{k+1}}{2} \|a_l^{k+1} - a_l^k\|_2^2).
\end{aligned} \tag{B.3}$$

So $F(\mathbf{W}^K, \mathbf{z}^K, \mathbf{a}^K) \leq F(\mathbf{W}^0, \mathbf{z}^0, \mathbf{a}^0)$. This proves the upper boundness of F. Let

$K \to \infty$ in Equation (B.3), since $F > 0$ is lower bounded, we have

$$\sum_{k=0}^{K}(\sum_{l=1}^{L}(\frac{\alpha_l^{k+1}}{2}\|W_l^{k+1}-W_l^k\|_2^2+\frac{\gamma_l^{k+1}}{2}\|z_l^{k+1}-z_l^k\|_2^2)+\sum_{l=1}^{L-1}\frac{\delta_l^{k+1}}{2}\|a_l^{k+1}-a_l^k\|_2^2) < \infty. \tag{B.4}$$

Since the sum of this infinite series is finite, every term converges to 0. This means that $\lim_{k\to\infty} W_l^{k+1}-W_l^k = 0$, $\lim_{k\to\infty} z_l^{k+1}-z_l^k = 0$ and $\lim_{k\to\infty} a_l^{k+1}-a_l^k = 0$. In other words, $\lim_{k\to\infty} \mathbf{W}^{k+1}-\mathbf{W}^k = 0$, $\lim_{k\to\infty} \mathbf{z}^{k+1}-\mathbf{z}^k = 0$, and $\lim_{k\to\infty} \mathbf{a}^{k+1}-\mathbf{a}^k = 0$.
(b). Because $F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is bounded, by the definition of coercivity and Assumption 4, $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is bounded.

□

Proof of Lemma 4

Proof. As shown in Remark 2.2 in [115],

$$\partial_{\mathbf{W}^{k+1}} F = \{\partial_{W_1^{k+1}} F\} \times \{\partial_{W_2^{k+1}} F\} \times \cdots \times \{\partial_{W_L^{k+1}} F\},$$

where $\times$ denotes Cartesian Product.
In Algorithm 4, for W_l^{k+1}, according to Line 6 of Algorithm 4, if $F(\mathbf{W}_{\le l}^{k+1}, \mathbf{z}_{\le l-1}^{k+1}, \mathbf{a}_{\le l-1}^{k+1}) < F(\mathbf{W}_{\le l-1}^{k+1}, \mathbf{z}_{\le l-1}^{k+1}, \mathbf{a}_{\le l-1}^{k+1})$, then

$$\begin{aligned}
\partial_{W_l^{k+1}} F &= \nabla_{W_l^{k+1}} \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^{k+1}) \quad \text{(Definition of } F \text{ in Problem 3)} \\
&= \nabla_{W_l^{k+1}} \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^{k+1}) - \nabla_{\overline{W}_l^{k+1}} \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) - \theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1}) \\
&+ \nabla_{\overline{W}_l^{k+1}} \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + \theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1}) \\
&= \rho(W_l^{k+1}-\overline{W}_l^{k+1}) a_{l-1}^{k+1}(a_{l-1}^{k+1})^T - \rho(z_l^{k+1}-z_l^k)(a_{l-1}^{k+1})^T - \theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1}) \\
&+ \nabla_{\overline{W}_l^{k+1}} \phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + \theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1}).
\end{aligned} \tag{B.5}$$

On one hand, we have

$$
\begin{aligned}
&\|\rho(W_l^{k+1}-\overline{W}_l^{k+1})a_{l-1}^{k+1}(a_{l-1}^{k+1})^T-\rho(z_l^{k+1}-z_l^k)(a_{l-1}^{k+1})^T-\theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1})\|\\
&\leq \rho\|(W_l^{k+1}-\overline{W}_l^{k+1})a_{l-1}^{k+1}(a_{l-1}^{k+1})^T\|+\rho\|(z_l^{k+1}-z_l^k)(a_{l-1}^{k+1})^T\|+\theta_l^{k+1}\|W_l^{k+1}-\overline{W}_l^{k+1}\|\\
&\text{(Triangle Inequality)}\\
&\leq \rho\|W_l^{k+1}-\overline{W}_l^{k+1}\|\|a_{l-1}^{k+1}\|\|a_{l-1}^{k+1}\|+\rho\|z_l^{k+1}-z_l^k\|\|a_{l-1}^{k+1}\|+\theta_l^{k+1}\|W_l^{k+1}-\overline{W}_l^{k+1}\|\\
&\text{(Cauchy-Schwarz Inequality)}\\
&\leq \rho M_{\mathbf{a}}\|z_l^{k+1}-z_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^{k+1}-\overline{W}_l^{k+1}\| \text{ (Lemma 3)} \qquad \text{(B.6)}\\
&\leq \rho M_{\mathbf{a}}\|z_l^{k+1}-z_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^{k+1}-(W_l^k+\omega^k(W_l^k-W_l^{k-1}))\|\\
&\text{(Nesterov Acceleration)}\\
&\leq \rho M_{\mathbf{a}}\|z_l^{k+1}-z_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^{k+1}-W_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^k-W_l^{k-1}\|\\
&\text{(Triangle Inequality and } \omega^k<1).
\end{aligned}
$$

On the other hand, Lemma 17 holds, that is

$$\nabla_{\overline{W}_l^{k+1}}\phi(a_{l-1}^{k+1},\overline{W}_l^{k+1},z_l^k)+\theta_l^{k+1}(W_l^{k+1}-\overline{W}_l^{k+1})=0.$$

Therefore, there exists $g_{1,l}^{k+1}\in\partial_{W_l^{k+1}}F$ such that

$$\|g_{1,l}^{k+1}\|\leq\rho M_{\mathbf{a}}\|z_l^{k+1}-z_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^{k+1}-W_l^k\|+(\rho M_{\mathbf{a}}^2+\theta_l^{k+1})\|W_l^k-W_l^{k-1}\|.$$

This shows that there exists $g_1^{k+1}=g_{1,1}^{k+1}\times g_{1,2}^{k+1}\times\cdots\times g_{1,L}^{k+1}\in\partial_{\mathbf{W}^{k+1}}F$ and $C_2=\max(\rho M_{\mathbf{a}},\rho M_{\mathbf{a}}^2+\theta_1^{k+1},\rho M_{\mathbf{a}}^2+\theta_2^{k+1},\cdots,\rho M_{\mathbf{a}}^2+\theta_L^{k+1})$ such that

$$\|g_l^{k+1}\|\leq C_2(\|\mathbf{W}^{k+1}-\mathbf{W}^k\|+\|\mathbf{z}^{k+1}-\mathbf{z}^k\|+\|\mathbf{W}^k-\mathbf{W}^{k-1}\|). \qquad \text{(B.7)}$$

Otherwise, we have

$$\begin{aligned}
&\|\rho(W_l^{k+1} - \overline{W}_l^{k+1})a_{l-1}^{k+1}(a_{l-1}^{k+1})^T - \rho(z_l^{k+1} - z_l^k)(a_{l-1}^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - \overline{W}_l^{k+1})\| \\
&\leq \rho M_{\mathbf{a}}\|z_l^{k+1} - z_l^k\| + (\rho M_{\mathbf{a}}^2 + \theta_l^{k+1})\|W_l^{k+1} - \overline{W}_l^{k+1}\|(\text{Equation (B.6)}) \\
&= \rho M_{\mathbf{a}}\|z_l^{k+1} - z_l^k\| + (\rho M_{\mathbf{a}}^2 + \theta_l^{k+1})\|W_l^{k+1} - W_l^k\|(\overline{W}_l^{k+1} = W_l^k).
\end{aligned}$$

Similarly, Lemma 17 holds, that is

$$\nabla_{\overline{W}_l^{k+1}}\phi(a_{l-1}^{k+1}, \overline{W}_l^{k+1}, z_l^k) + \theta_l^{k+1}(W_l^{k+1} - \overline{W}_l^{k+1}) = 0.$$

By Equation (B.5), we know that there exists $g_{1,l}^{k+1} \in \partial_{W_l^{k+1}} F$ such that

$$\|g_{1,l}^{k+1}\| \leq \rho M_{\mathbf{a}}\|z_l^{k+1} - z_l^k\| + (\rho M_{\mathbf{a}}^2 + \theta_l^{k+1})\|W_l^{k+1} - W_l^k\|. \tag{B.8}$$

Combining Equation (B.7) with Equation (B.8), we show that there exists $g_1^{k+1} = g_{1,1}^{k+1} \times g_{1,2}^{k+1} \times \cdots \times g_{1,L}^{k+1} \in \partial_{\mathbf{W}^{k+1}} F$ and $C_2 = \max(\rho M_{\mathbf{a}}, \rho M_{\mathbf{a}}^2 + \theta_1^{k+1}, \rho M_{\mathbf{a}}^2 + \theta_2^{k+1}, \cdots, \rho M_{\mathbf{a}}^2 + \theta_L^{k+1})$ such that

$$\|g_l^{k+1}\| \leq C_2(\|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\| + \|\mathbf{W}^k - \mathbf{W}^{k-1}\|).$$

□

Proof of Lemma 2

Proof. We add Equation (3.7), Equation (3.8), and Equation (3.9) from $l = 1$ to L to obtain

$$\begin{aligned}
&F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) - F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) \\
&\geq \sum\nolimits_{l=1}^{L}(\frac{\alpha_l^{k+1}}{2}\|W_l^{k+1} - W_l^k\|_2^2 + \frac{\gamma_l^{k+1}}{2}\|z_l^{k+1} - z_l^k\|_2^2) + \sum\nolimits_{l=1}^{L-1}\frac{\delta_l^{k+1}}{2}\|a_l^{k+1} - a_l^k\|_2^2.
\end{aligned}$$

Let $C_5 = \min(\frac{\alpha_l^{k+1}}{2}, \frac{\gamma_l^{k+1}}{2}, \frac{\delta_l^{k+1}}{2}) > 0$, we have

$$\begin{aligned}
&F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) - F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) \\
&\geq C_5(\sum\nolimits_{l=1}^{L}(\|W_l^{k+1} - W_l^k\|_2^2 + \|z_l^{k+1} - z_l^k\|_2^2) + \sum\nolimits_{l=1}^{L-1}\|a_l^{k+1} - a_l^k\|_2^2) \\
&= C_5(\|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 + \|\mathbf{a}^{k+1} - \mathbf{a}^k\|_2^2) \qquad \text{(B.9)} \\
&\geq 0.
\end{aligned}$$

By Lemma 3(b) and a monotone sequence is convergent if it is bounded, then $F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k)$ is convergent. □

Proof of Theorem 4

Proof. By Lemma 3 (a), $\lim_{k\to\infty} \mathbf{W}^{k+1} - \mathbf{W}^k = 0$. By Lemma 3 (b), there exists a subsequence $\mathbf{W}^s$ such that $\mathbf{W}^s \to \mathbf{W}^*$, where W^* is a limit point. From Lemma 4, there exist $g_1^s \in \partial_{\mathbf{W}^s} F$ such that $\|g_1^s\| \to 0$ as $s \to \infty$. According to the definition of limiting subdifferential, we have $0 \in \partial_{\mathbf{W}^*} F$. In other words, $\mathbf{W}^*$ is a stationary point of F in Problem 3. □

Proof of Theorem 5

Proof. In Algorithm 4, we prove this by the KL Property.
Firstly, we consider Equation (3.4) and Equation (3.6), by Lemma 3, $f_l(\overline{z}_l^{k+1} - \nabla\phi_{\overline{z}_l^{k+1}}/\rho) - a_l^k$ and $f_l(z^{k+1}) - \overline{a}_l^{k+1} + \nabla_{\overline{a}_l^{k+1}}\phi/\tau_l^{k+1}$ are bounded, i.e. there exist constants D_1 and D_2 such that

$$\begin{aligned}
&| f_l(\overline{z}_l^{k+1} - \nabla_{\overline{z}_l^{k+1}}\phi/\rho) - a_l^k | < D_1. \\
&| f_l(z^{k+1}) - \overline{a}_l^{k+1} + \nabla_{\overline{a}_l^{k+1}}\phi/\tau_l^{k+1} | < D_2.
\end{aligned}$$

Let $\varepsilon = \max(D_1, D_2)$, then the solutions to Equation (3.4) and Equation (3.6) are

simplified as follows:

$$z_l^{k+1} \leftarrow \overline{z}_l^{k+1} - \nabla_{\overline{z}_l^{k+1}} \phi / \rho. \tag{B.10}$$

$$a_l^{k+1} \leftarrow \overline{a}_l^{k+1} - \nabla_{\overline{a}_l^{k+1}} \phi / \tau_l^{k+1}. \tag{B.11}$$

This is because $f_l(z_l^{k+1}) - \varepsilon \leq a_l^k \leq f_l(z_l^{k+1}) + \varepsilon$ and $f_l(z_l^{k+1}) - \varepsilon \leq a_l^{k+1} \leq f_l(z_l^{k+1}) + \varepsilon$ hold in Equation (3.4) and Equation (3.6), respectively.

Next, we prove that given $\varepsilon = \max(D_1, D_2)$, there exists $C_3 = \max(\rho M_{\mathbf{W}}^2 + \tau_1^{k+1}, \rho M_{\mathbf{W}}^2 + \tau_2^{k+1}, \rho M_{\mathbf{W}}^2 + \tau_3^{k+1}, \cdots, \rho M_{\mathbf{W}}^2 + \tau_{L-1}^{k+1}, 2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})$, some $g_3^{k+1} \in \partial_{\mathbf{z}^{k+1}} F$ and $g_4^{k+1} \in \partial_{\mathbf{a}^{k+1}} F$ such that

$$\|g_3^{k+1}\| = 0,$$
$$\|g_4^{k+1}\| \leq C_3(\|\mathbf{a}^{k+1} - \mathbf{a}^k\| + \|\mathbf{a}^k - \mathbf{a}^{k-1}\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|).$$

As shown in [107, 115],

$$\partial_{\mathbf{z}^{k+1}} F = \partial_{z_1^{k+1}} F \times \partial_{z_2^{k+1}} F \times \cdots \times \partial_{z_L^{k+1}} F,$$
$$\nabla_{\mathbf{a}^{k+1}} F = \nabla_{a_1^{k+1}} F \times \nabla_{a_2^{k+1}} F \times \cdots \times \nabla_{a_{L-1}^{k+1}} F,$$

where $\times$ denotes Cartesian Product.

For $z_l^{k+1} (l < L)$, according to Line 18 of Algorithm 4, no matter

$F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1}) \geq F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l-1}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$ or not, we have

$$\begin{aligned}
\partial_{z_l^{k+1}} F &= \nabla_{z_l^{k+1}} \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^{k+1}) \\
&= \nabla_{z_l^{k+1}} \phi(a_{l-1}^{k+1}, W_l^{k+1}, z_l^{k+1}) - \nabla_{\overline{z}_l^{k+1}} \phi(a_{l-1}^{k+1}, W_l^{k+1}, \overline{z}_l^{k+1}) - \rho(z_l^{k+1} - \overline{z}_l^{k+1}) \\
&\quad \text{(Equation (B.10))} \\
&= 0.
\end{aligned}$$

For z_L^{k+1}, according to Line 12 of Algorithm 4, no matter
$F(\mathbf{W}_{\leq L}^{k+1}, \mathbf{z}_{\leq L}^{k+1}, \mathbf{a}_{\leq L-1}^{k+1}) \geq F(\mathbf{W}_{\leq L}^{k+1}, \mathbf{z}_{\leq L-1}^{k+1}, \mathbf{a}_{\leq L-1}^{k+1})$ or not, we have

$$\begin{aligned}
\partial_{z_L^{k+1}} F &= \nabla_{z_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, z_L^{k+1}) + \partial R(z_L^{k+1}; y) \\
&= \nabla_{z_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, z_L^{k+1}) + \partial R(z_L^{k+1}; y) + \nabla_{\overline{z}_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, \overline{z}_L^{k+1}) \\
&\quad + \rho(z_L - \overline{z}_L^{k+1}) - \nabla_{\overline{z}_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, \overline{z}_L^{k+1}) - \rho(z_L^{k+1} - \overline{z}_L^{k+1}) \\
&= \nabla_{z_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, z_L^{k+1}) - \nabla_{\overline{z}_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, \overline{z}_L^{k+1}) - \rho(z_L^{k+1} - \overline{z}_L^{k+1}) \\
&\quad (0 \in \partial R(z_L^{k+1}; y) + \nabla_{\overline{z}_L^{k+1}} \phi(a_{L-1}^{k+1}, W_L^{k+1}, \overline{z}_L^{k+1}) + \rho(z_L^{k+1} - \overline{z}_L^{k+1}) \\
&\quad \text{by the optimality condition of Equation (3.5))} \\
&= 0.
\end{aligned}$$

Therefore, there exists $g_{3,l}^{k+1} = \nabla_{z_l^{k+1}} F$ such that $\|g_{3,l}^{k+1}\| = 0$. This shows that there exists $g_3^{k+1} = g_{3,1}^{k+1} \times g_{3,2}^{k+1} \times \cdots \times g_{3,L}^{k+1} = \nabla_{\mathbf{z}^{k+1}} F$ such that

$$\|g_3^{k+1}\| = 0. \tag{B.12}$$

For a_l^{k+1}, we have

$$
\begin{aligned}
\partial_{a_l^{k+1}} F &= \nabla_{a_l^{k+1}} \phi(a_l^{k+1}, W_{l+1}^k, z_{l+1}^{k+1}) \\
&= \nabla_{a_l^{k+1}} \phi(a_l^{k+1}, W_{l+1}^{k+1}, z_{l+1}^{k+1}) - \nabla_{\overline{a}_l^{k+1}} \phi(\overline{a}_l^{k+1}, W_{l+1}^k, z_{l+1}^k) - \tau_l^{k+1}(a_l^{k+1} - \overline{a}_l^{k+1}) \\
&\quad \text{(Equation (B.11))} \\
&= \rho(W_{l+1}^{k+1})^T(W_{l+1}^{k+1} a_l^{k+1} - z_{l+1}^{k+1}) - \rho(W_{l+1}^k)^T(W_{l+1}^k \overline{a}_l^{k+1} - z_{l+1}^k) \\
&\quad - \tau_l^{k+1}(a_l^{k+1} - \overline{a}_l^{k+1}) \\
&= \rho(W_{l+1}^{k+1})^T W_{l+1}^{k+1}(a_l^{k+1} - \overline{a}_l^{k+1}) + \rho(W_{l+1}^{k+1})^T(W_{l+1}^{k+1} - W_{l+1}^k)\overline{a}_l^{k+1} \\
&\quad + \rho(W_{l+1}^{k+1} - W_{l+1}^k)^T W_{l+1}^k \overline{a}_l^{k+1} - \rho(W_{l+1}^{k+1})^T(z_{l+1}^{k+1} - z_{l+1}^k) - \rho(W_{l+1}^{k+1} - W_{l+1}^k)^T z_{l+1}^k \\
&\quad - \tau_l^{k+1}(a_l^{k+1} - \overline{a}_l^{k+1}).
\end{aligned}
$$

Therefore

$$
\begin{aligned}
\|\partial_{a_l^{k+1}} F\| &\leq \rho\|W_{l+1}^{k+1}\|\|W_{l+1}^{k+1}\|\|a_l^{k+1} - \overline{a}_l^{k+1}\| + \rho\|W_{l+1}^{k+1}\|\|W_{l+1}^{k+1} - W_{l+1}^k\|\|\overline{a}_l^{k+1}\| \\
&\quad + \rho\|W_{l+1}^{k+1} - W_{l+1}^k\|\|W_{l+1}^k\|\|\overline{a}_l^{k+1}\| + \rho\|W_{l+1}^{k+1}\|\|z_{l+1}^{k+1} - z_{l+1}^k\| \\
&\quad + \rho\|W_{l+1}^{k+1} - W_{l+1}^k\|\|z_{l+1}^k\| + \tau_l^{k+1}\|a_l^{k+1} - \overline{a}_l^{k+1}\| \\
&\quad \text{(Triangle Inequality and Cauthy-Schwarz Inequality)} \\
&\quad \leq \rho M_{\mathbf{W}}^2 \|a_l^{k+1} - \overline{a}_l^{k+1}\| \\
&\quad + \rho M_{\mathbf{W}}\|W_{l+1}^{k+1} - W_{l+1}^k\| M_{\mathbf{a}} + \rho\|W_{l+1}^{k+1} - W_{l+1}^k\| M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\| \\
&\quad + \rho\|W_{l+1}^{k+1} - W_{l+1}^k\| M_{\mathbf{z}} + \tau_l^{k+1}\|a_l^{k+1} - \overline{a}_l^{k+1}\| \text{ (Lemma 3)} \\
&\quad = (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - \overline{a}_l^{k+1}\| + (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| \\
&\quad + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\|.
\end{aligned}
$$

According to Line 22 of Algorithm 4, if

$F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l}^{k+1}) < F(\mathbf{W}_{\leq l}^{k+1}, \mathbf{z}_{\leq l}^{k+1}, \mathbf{a}_{\leq l-1}^{k+1})$, then we have

$$\begin{aligned}
\|\partial_{a_l^{k+1}} F\| &\leq (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - a_l^k - (a_l^k - a_l^{k-1})\omega^k\| \\
&\quad + (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\| \\
&\quad \text{(Nestrov Acceleration)} \\
&\quad \leq (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - a_l^k\| + (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^k - a_l^{k-1}\| + \\
&\quad (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\| \\
&\quad \text{(Triangle Inequality and } \omega^k < 1).
\end{aligned}$$

Therefore, there exists $g_{4,l}^{k+1} \in \partial_{a_l^{k+1}} F$ such that

$$\begin{aligned}
\|g_{4,l}^{k+1}\| &\leq (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - a_l^k\| + (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^k - a_l^{k-1}\| \\
&\quad + (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\|. \qquad \text{(B.13)}
\end{aligned}$$

Otherwise,

$$\begin{aligned}
\|\partial_{a_l^{k+1}} F\| &\leq (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - a_l^k\| + (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| \\
&\quad + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\| \ (\overline{a}_l^{k+1} = a_l^k).
\end{aligned}$$

Therefore, there exists $g_{4,l}^{k+1} \in \partial_{a_l^{k+1}} F$ such that

$$\begin{aligned}
\|g_{4,l}^{k+1}\| &\leq (\rho M_{\mathbf{W}}^2 + \tau_l^{k+1})\|a_l^{k+1} - a_l^k\| + (2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})\|W_{l+1}^{k+1} - W_{l+1}^k\| \\
&\quad + \rho M_{\mathbf{W}}\|z_{l+1}^{k+1} - z_{l+1}^k\|. \qquad \text{(B.14)}
\end{aligned}$$

Combining Equation (B.13) and Equation (B.14), we show that there exists $g_4^{k+1} = g_{4,1}^{k+1} \times g_{4,2}^{k+1} \times \cdots \times g_{4,L}^{k+1} \in \partial_{\mathbf{a}^{k+1}} F$ and $C_3 = \max(\rho M_{\mathbf{W}}^2 + \tau_1^{k+1}, \rho M_{\mathbf{W}}^2 + \tau_2^{k+1}, \rho M_{\mathbf{W}}^2 +$

$\tau_3^{k+1}, \cdots, \rho M_{\mathbf{W}}^2 + \tau_{L-1}^{k+1}, 2\rho M_{\mathbf{W}} M_{\mathbf{a}} + \rho M_{\mathbf{z}})$ such that

$$\|g_4^{k+1}\| \leq C_3(\|\mathbf{a}^{k+1} - \mathbf{a}^k\| + \|\mathbf{a}^k - \mathbf{a}^{k-1}\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|). \quad \text{(B.15)}$$

Combining Lemma 4, Equation (B.12) and Equation (B.15), we prove that there exists $g^{k+1} \in \partial F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) = \{\partial_{\mathbf{W}^{k+1}} F, \partial_{\mathbf{z}^{k+1}} F, \partial_{\mathbf{a}^{k+1}} F\}$ and $C_4 = \max(C_2, C_3, \rho)$ such that

$$\begin{aligned}\|g^{k+1}\| \leq C_4(&\|\mathbf{a}^{k+1} - \mathbf{a}^k\| + \|\mathbf{a}^k - \mathbf{a}^{k-1}\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| \\ &+ \|\mathbf{W}^k - \mathbf{W}^{k-1}\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|). \end{aligned} \quad \text{(B.16)}$$

Finally, we prove the linear convergence rate by the KL Property given Equation (B.16) and Equation (B.9). Because F is locally strongly convex with a constant μ, F satisfies the KL Property by Lemma 16. Let $F^* = F(\mathbf{W}^*, \mathbf{z}^*, \mathbf{a}^*)$ be the convergent value of F, by Lemma 2, $F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) \to F^*$, then for any $\eta_1 > 0$ there exists $k_2 \in \mathbb{N}$ such that it holds for $k > k_2$ that $F^* < F(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) < F^* + \eta_1$. Also by Lemma 3(a) and Equation (B.16), $g^{k+1} \to 0$ as $k \to \infty$, then for any $\eta_2 > 0$ there exists $k_3 \in \mathbb{N}$, such that it holds for $k > k_3$ that $\|g^{k+1}\| < \eta_2$. Therefore, for any $k > k_1 = \max(k_2, k_3)$, $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{a}^k) \in \{(\mathbf{W}, \mathbf{z}, \mathbf{a}) : |F^* < F(\mathbf{W}, \mathbf{z}, \mathbf{a}) < F^* + \eta_1 \cap \exists g \in$

$F(\mathbf{W}, \mathbf{z}, \mathbf{a})$ *s.t.* $\|g\| < \eta_2\}$. By the KL Property and Lemma 16, it holds that

$$
\begin{aligned}
1 &\leq \|g^{k+1}\|/(\mu\sqrt{F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*}) \\
&\leq C_4(\|\mathbf{a}^{k+1} - \mathbf{a}^k\| + \|\mathbf{a}^k - \mathbf{a}^{k-1}\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{W}^k - \mathbf{W}^{k-1}\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|) \\
&/(\mu\sqrt{F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*}) \text{ (Equation (B.16))} \\
&\leq C_4^2(\|\mathbf{a}^{k+1} - \mathbf{a}^k\| + \|\mathbf{a}^k - \mathbf{a}^{k-1}\| + \|\mathbf{W}^{k+1} - \mathbf{W}^k\| + \|\mathbf{W}^k - \mathbf{W}^{k-1}\| + \|\mathbf{z}^{k+1} - \mathbf{z}^k\|)^2 \\
&/(\mu^2(F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*)) \\
&\leq (5C_4^2(\|\mathbf{a}^{k+1} - \mathbf{a}^k\|_2^2 + \|\mathbf{a}^k - \mathbf{a}^{k-1}\|_2^2 + \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 + \|\mathbf{W}^k - \mathbf{W}^{k-1}\|_2^2 \\
&+ \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2))/(\mu^2(F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*))\text{(Mean Inequality)} \\
&\leq (5C_4^2(F(\mathbf{W}^{k-1}, \mathbf{z}^{k-1}, \mathbf{a}^{k-1}) - F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}))) \\
&/(C_5\mu^2(F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*)) \text{ (Equation (B.9))}.
\end{aligned}
$$

This indicates that

$$(C_5\mu^2 + 5C_4^2)(F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^*) \leq 5C_4^2(F(\mathbf{W}^{k-1}, \mathbf{z}^{k-1}, \mathbf{a}^{k-1}) - F^*).$$

Let $0 < C_1 = \frac{5C_4^2}{C_5\mu^2 + 5C_4^2} < 1$, we have

$$F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^* \leq C_1(F(\mathbf{W}^{k-1}, \mathbf{z}^{k-1}, \mathbf{a}^{k-1}) - F^*).$$

So in summary, for any ρ, there exist $\varepsilon = \max(D_1, D_2)$, $k_1 = \max(k_2, k_3)$, and $0 < C_1 = \frac{5C_4^2}{C_5\mu^2 + 5C_4^2} < 1$ such that

$$F(\mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{a}^{k+1}) - F^* \leq C_1(F(\mathbf{W}^{k-1}, \mathbf{z}^{k-1}, \mathbf{a}^{k-1}) - F^*).$$

for $k > k_1$. In other words, the linear convergence rate is proven. □

Appendix C

Appendix of the pdADMM Algorithm

C.1 Preliminary Results

Lemma 19. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$u_l^k = \nu(q_l^k - f_l(z_l^k))$$

Proof. This follows directly from the optimality condition of q_l^k and Equation (4.6). □

Lemma 20. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$\|u_l^{k+1} - u_l^k\| \leq \nu\|q_l^{k+1} - q_l^k\| + \nu S\|z_l^{k+1} - z_l^k\|$$

Proof.

$$\begin{aligned}
&\|u_l^{k+1} - u_l^k\| \\
&= \|\nu(q_l^{k+1} - f_l(z_l^{k+1})) - \nu(q_l^k - f_l(z_l^k))\| \text{(Lemma 19)} \\
&\leq \nu\|q_l^{k+1} - q_l^k\| + \nu\|f_l(z_l^{k+1}) - f_l(z_l^k)\| \text{(Triangle Inequality)} \\
&\leq \nu\|q_l^{k+1} - q_l^k\| + \nu S\|z_l^{k+1} - z_l^k\| \text{(Assumption 5)}
\end{aligned}$$

□

Lemma 21. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$\|u_l^{k+1} - u_l^k\|_2^2 \leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + S^2\|z_l^{k+1} - z_l^k\|_2^2)$$

Proof.

$$\begin{aligned}
\|u_l^{k+1} - u_l^k\|_2^2 &= \nu^2\|q_l^{k+1} - f_l(z_l^{k+1}) - q_l^k + f_l(z_l^k)\|_2^2 \text{(Lemma 19)} \\
&\leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + \|f_l(z_l^{k+1}) - f_l(z_l^k)\|_2^2) \text{(Mean Inequality)} \\
&\leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + S^2\|z_l^{k+1} - z_l^k\|_2^2) \text{(Assumption 5)}
\end{aligned}$$

□

Lemma 22. For every $k \in \mathbb{N}$, it holds that

$$L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \geq \sum\nolimits_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 \tag{C.1}$$

$$\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&\geq \sum\nolimits_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1} - W_l^k\|_2^2
\end{aligned} \tag{C.2}$$

$$\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) \\
&\geq (\nu/2)\sum\nolimits_{l=1}^{L}\|z_l^{k+1} - z_l^k\|_2^2
\end{aligned} \tag{C.3}$$

Proof. Generally, all inequalities can be obtained by applying optimality conditions of updating $\mathbf{p}$, $\mathbf{W}$, and $\mathbf{z}$, respectively. We only prove Inequalities (C.1), and (C.3). This is because Inequality (C.2) follows the same routine of Inequality (C.1).

Firstly, we focus on Inequality (C.1). The choice of τ_l^{k+1} requires

$$\phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) \leq U_l(p_l^{k+1}; \tau_l^{k+1}) \tag{C.4}$$

Moreover, the optimality condition of Equation (4.1) leads to

$$\nabla_{p_l^k}\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + \tau_l^{k+1}(p_l^{k+1} - p_l^k) = 0 \tag{C.5}$$

Therefore

$$\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&= \sum\nolimits_{l=2}^{L}(\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) - \phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)) \\
&\geq \sum\nolimits_{l=2}^{L}(\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) - U_l(p_l^{k+1}; \tau_l^{k+1}))(\text{Inequality (C.4)}) \\
&= \sum\nolimits_{l=2}^{L}(-\nabla_{p_l^k}\phi^T(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)(p_l^{k+1} - p_l^k) - (\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2) \\
&= \sum\nolimits_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 \quad (\text{Equation (C.5)})
\end{aligned}$$

Next, we prove Inequality (C.3). Because z_l^{k+1} minimizes Equation (4.3), we have

$$\begin{aligned}
&(\nu/2)\|z_l^{k+1} - W_l^{k+1}p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^{k+1})\|_2^2 + (\nu/2)\|z_l^{k+1} - z_l^k\|_2^2 \\
&\leq (\nu/2)\|z_l^k - W_l^{k+1}p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^k)\|_2^2
\end{aligned} \tag{C.6}$$

And

$$
\begin{aligned}
&R(z_L^k; y) + (\nu/2)\|z_L^k - W_L^{k+1} p_L^{k+1}\|_2^2 - R(z_L^{k+1}; y) - (\nu/2)\|z_L^{k+1} - W_L^{k+1} p_L^{k+1}\|_2^2 \\
&= R(z_L^k; y) - R(z_L^{k+1}; y) + (\nu/2)\|z_L^k - z_L^{k+1}\|_2^2 + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1})^T (z_L^k - z_L^{k+1}) \\
&(\|a-b\|_2^2 - \|a-c\|_2^2 = \|b-c\|_2^2 + 2(c-a)^T(b-c) \text{ where} \\
&a = W_L^{k+1} p_L^{k+1},\ b = z_L^k, \text{ and } c = z_L^{k+1}) \qquad \text{(C.7)} \\
&\geq s^T (z_L^k - z_L^{k+1}) + (\nu/2)\|z_L^k - z_L^{k+1}\|_2^2 + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1})^T (z_L^k - z_L^{k+1}) \\
&(s \in \partial R(z_L^{k+1}; y) \text{ is a subgradient of } R(z_L^{k+1}; y)) \\
&= (\nu/2)\|z_L^{k+1} - z_L^k\|_2^2 \qquad \text{(C.8)} \\
&(0 \in s + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1}) \text{ by the optimality condition of Equation (4.4))}
\end{aligned}
$$

Therefore

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) \\
&= \sum\nolimits_{i=1}^{L-1} ((\nu/2)\|z_l^k - W_l^{k+1} p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^k)\|_2^2 \\
&- (\nu/2)\|z_l^{k+1} - W_l^{k+1} p_l^{k+1}\|_2^2 - (\nu/2)\|q_l^k - f_l(z_l^{k+1})\|_2^2) \\
&+ R(z_L^k; y) + (\nu/2)\|z_L^k - W_L^{k+1} p_L^{k+1}\|_2^2 - R(z_L^{k+1}; y) - (\nu/2)\|z_L^{k+1} - W_L^{k+1} p_L^{k+1}\|_2^2 \\
&\geq (\nu/2) \sum\nolimits_{l=1}^{L} \|z_l^{k+1} - z_l^k\|_2^2 \quad \text{(Inequalities (C.6) and (C.8))}
\end{aligned}
$$

□

Lemma 23. For every $k \in N$, it holds that

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \sum\nolimits_{l=1}^{L-1} ((\rho/2 - 2\nu^2/\rho - \nu/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2) \qquad \text{(C.9)}
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \sum_{l=1}^{L-1}((\nu/2)\|f_l(z_l^{k+1}) - q_l^k\|_2^2 - (\nu/2)\|f_l(z_l^{k+1}) - q_l^{k+1}\|_2^2 - (u_l^{k+1})^T(q_l^k - q_l^{k+1}) \\
&+ (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&= \sum_{l=1}^{L-1}((\nu/2)\|f_l(z_l^{k+1}) - q_l^k\|_2^2 - (\nu/2)\|f_l(z_l^{k+1}) - q_l^{k+1}\|_2^2 \\
&- \nu(q_l^{k+1} - f_l(z_l^{k+1}))^T(q_l^k - q_l^{k+1}) + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&\text{(Lemma 19)} \\
&\geq \sum_{l=1}^{L-1}(-(\nu/2)\|q_l^{k+1} - q_l^k\|_2^2 + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&(-\nu(q_l - f_l(z_l^{k+1})) = -(\nu/2)\nabla_{q_l}\|q_l - f_l(z_l^{k+1})\|_2^2 \text{ is lipschitz continuous} \\
&\text{with regard to } q_l \text{ and Lemma 2.1 in [2])} \\
&\geq \sum_{l=1}^{L-1}(-(\nu/2)\|q_l^{k+1} - q_l^k\|_2^2 + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2/\rho)\|q_l^{k+1} - q_l^k\|_2^2 \\
&- (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2) \text{ (Lemma 21)} \\
&= \sum_{l=1}^{L-1}((\rho/2 - 2\nu^2/\rho - \nu/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2)
\end{aligned}
$$

□

C.2 Main Proofs

Proof of Lemma 5

Proof. We sum up Inequalities (C.1), (C.2), (C.3), and (C.9) to obtain Inequality (4.7). □

Proof of Lemma 6

Proof. There exists $\mathbf{q}^{'}$ such that $p_{l+1}^k = q_l^{'}$ and

$$F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'}) \geq \min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}}\{F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}) | p_{l+1} = q_l\} > -\infty$$

Therefore, we have

$$\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&= F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k) + \sum\nolimits_{l=1}^{L} (u_l^k)^T (p_{l+1}^k - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2 \\
&= R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^k - f_l(z_l^k)\|_2^2) \\
&+ \sum\nolimits_{l=1}^{L-1} ((u_l^k)^T (p_{l+1}^k - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&= R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^k - f_l(z_l^k)\|_2^2) \\
&+ \sum\nolimits_{l=1}^{L-1} (\nu(q_l^k - f_l(z_l^k))^T (q_l^{'} - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&(p_{l+1}^k = q_l^{'} \text{ and Lemma 19}) \\
&\geq R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^{'} - f_l(z_l^k)\|_2^2) \\
&- \sum\nolimits_{l=1}^{L-1} (\nu/2)\|q_l^{'} - q_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&(\nu(q_l - f_l(z_l^{k+1})) = (\nu/2)\nabla_{q_l}\|q_l - f_l(z_l^{k+1})\|_2^2 \text{ is lipschitz continuous} \\
&\text{with regard to } q_l \text{ and Lemma 2.1 in [2])} \\
&= F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'}) + (\rho - \nu)/2\|p_{l+1}^k - q_l^k\|_2^2 > -\infty
\end{aligned}$$

Therefore, $F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'})$ and $(\rho - \nu)/2\|p_{l+1}^k - q_l^k\|_2^2$ are upper bounded by $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ and hence $L_\rho(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$ (Lemma 5). From Assumption 5, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k)$ is bounded. $\mathbf{q}^k$ is also bounded because $(\rho - \nu)/2\|p_{l+1}^k - q_l^k\|_2^2$ is upper bounded. $\mathbf{u}^k$ is bounded because of Lemma 19. □

Proof of Lemma 7

Proof. We know that $\partial L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$
$= \{\nabla_{\mathrm{p}^{k+1}} L_\rho, \nabla_{\mathrm{W}^{k+1}} L_\rho, \partial_{\mathrm{z}^{k+1}} L_\rho, \nabla_{\mathrm{q}^{k+1}} L_\rho, \nabla_{\mathrm{u}^{k+1}} L_\rho\}$ [101]. Specifically, we prove that

$\|g\|$ is upper bounded by the linear combination of $\|\mathbf{p}^{k+1}-\mathbf{p}^k\|$,$\|\mathbf{W}^{k+1}-\mathbf{W}^k\|$, $\|\mathbf{z}^{k+1}-\mathbf{z}^k\|$, $\|\mathbf{q}^{k+1}-\mathbf{q}^k\|$, and $\|\mathbf{u}^{k+1}-\mathbf{u}^k\|$.

For p_l^{k+1},

$$
\begin{aligned}
&\nabla_{p_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{p_l^{k+1}} \phi(p_l^{k+1}, W_l^{k+1}, z_l^{k+1}, q_{l-1}^{k+1}, u_{l-1}^{k+1}) \\
&= \nabla_{p_l^k} \phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + \tau_l^{k+1}(p_l^{k+1} - p_l^k) - \tau_l^{k+1}(p_l^{k+1} - p_l^k) \\
&+ \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k - \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k \\
&+ (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k) \\
&= -\tau_l^{k+1}(p_l^{k+1} - p_l^k) + \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k \\
&- \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k + (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k)
\end{aligned}
$$

(The optimality condition of Equation (4.1))

So

$$
\begin{aligned}
&\|\nabla_{p_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \|\tau_l^{k+1}(p_l^{k+1} - p_l^k) + \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k \\
&- \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k + (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k)\| \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - (W_l^k)^T W_l^k p_l^k\| \\
&+ \nu\|(W_l^{k+1})^T z_l^{k+1} - (W_l^k)^T z_l^k\| + \|u_{l-1}^{k+1} - u_{l-1}^k\| + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\text{(Triangle Inequality)} \\
&= \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|(W_l^{k+1})^T W_l^{k+1}(p_l^{k+1} - p_l^k) + (W_l^{k+1})^T(W_l^{k+1} - W_l^k) p_l^k \\
&+ (W_l^{k+1} - W_l^k)^T W_l^k p_l^k\| + \nu\|(W_l^{k+1})^T(z_l^{k+1} - z_l^k) + (W_l^{k+1} - W_l^k)^T z_l^k\| \\
&+ \|u_{l-1}^{k+1} - u_{l-1}^k\| + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|W_l^{k+1}\|^2\|p_l^{k+1} - p_l^k\| + \nu\|W_l^{k+1}\|\|W_l^{k+1} - W_l^k\|\|p_l^k\| \\
&+ \nu\|W_l^{k+1} - W_l^k\|\|W_l^k\|\|p_l^k\| + \nu\|W_l^{k+1}\|\|z_l^{k+1} - z_l^k\| + \nu\|W_l^{k+1} - W_l^k\|\|z_l^k\| \\
&+ \nu(\|q_{l-1}^{k+1} - q_{l-1}^k\| + S\|z_{l-1}^{k+1} - z_{l-1}^k\|) + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\text{(Triangle Inequality, Cauthy-Schwartz Inequality and Lemma 20)} \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\mathbb{N}_{\mathbf{W}}^2\|p_l^{k+1} - p_l^k\| + 2\nu\mathbb{N}_{\mathbf{W}}\mathbb{N}_{\mathbf{p}}\|W_l^{k+1} - W_l^k\| \\
&+ \nu\mathbb{N}_{\mathbf{W}}\|z_l^{k+1} - z_l^k\| + \nu\mathbb{N}_{\mathbf{z}}\|W_l^{k+1} - W_l^k\| \\
&+ 2\nu^2(\|q_{l-1}^{k+1} - q_{l-1}^k\|_2^2 + S^2\|z_{l-1}^{k+1} - z_{l-1}^k\|_2^2) + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\text{(Lemma 6)}
\end{aligned}
$$

For W_1^{k+1},

$$\begin{aligned}
&\nabla_{W_1^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{W_1^{k+1}} \phi(p_1^{k+1}, W_1^{k+1}, z_1^{k+1}) \\
&= \nabla_{W_1^k} \phi(p_1^{k+1}, W_1^k, z_1^k) + \theta_1^{k+1}(W_1^{k+1} - W_1^k) + \nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T \\
&- \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k) \\
&= \nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T - \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k)
\end{aligned}$$

(The optimality condition of Equation (4.2))

So

$$\begin{aligned}
&\|\nabla_{W_1^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \|\nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T - \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k)\| \\
&\leq \nu\|W_1^{k+1} - W_1^k\|\|p_1^{k+1}\|^2 + \nu\|z_1^{k+1} - z_1^k\|\|p_1^{k+1}\| + \theta_1^{k+1}\|W_1^{k+1} - W_1^k\|
\end{aligned}$$

(Triangle Inequality and Cauthy-Schwartz Inequality)

$$\leq \nu\|W_1^{k+1} - W_1^k\|\mathbb{N}_{\mathbf{p}}^2 + \nu\|z_1^{k+1} - z_1^k\|\mathbb{N}_{\mathbf{p}} + \theta_1^{k+1}\|W_1^{k+1} - W_1^k\| \quad \text{(Theorem 6)}$$

For $W_l^{k+1} (1 < l \leq L)$,

$$\begin{aligned}
&\nabla_{W_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{W_l^{k+1}} \phi(p_l^{k+1}, W_l^{k+1}, z_l^{k+1}, p_{l-1}^{k+1}, u_{l-1}^{k+1}) \\
&= \nabla_{W_l^k} \phi(p_l^{k+1}, W_l^k, z_l^k, p_{l-1}^k, u_{l-1}^k) + \theta_l^{k+1}(W_l^{k+1} - W_1^k) + \nu(W_l^{k+1} - W_l^k) p_l^{k+1}(p_l^{k+1})^T \\
&- \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k) \\
&= \nu(W_l^{k+1} - W_l^k) p_l^{k+1}(p_l^{k+1})^T - \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k)
\end{aligned}$$

(The optimality condition of Equation (4.2))

So

$$
\begin{aligned}
&\|\nabla_{W_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \|\nu(W_l^{k+1} - W_l^k)p_l^{k+1}(p_l^{k+1})^T - \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k)\| \\
&\leq \nu\|W_l^{k+1} - W_l^k\|\|p_l^{k+1}\|^2 + \nu\|z_l^{k+1} - z_l^k\|\|p_l^{k+1}\| + \theta_l^{k+1}\|W_l^{k+1} - W_l^k\| \\
&\text{(Triangle Inequality and Cauthy-Schwartz Inequality)} \\
&\leq \nu\|W_l^{k+1} - W_l^k\|\mathbb{N}_{\mathbf{p}}^2 + \nu\|z_l^{k+1} - z_l^k\|\mathbb{N}_{\mathbf{p}} + \theta_l^{k+1}\|W_l^{k+1} - W_l^k\| \quad \text{(Theorem 6)}
\end{aligned}
$$

For $z_l^{k+1}(l < L)$,

$$
\begin{aligned}
&\partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) + \nu(z_l^{k+1} - z_l^k) - \nu(z_l^{k+1} - z_l^k) \\
&- \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k) \quad (\circ \text{ is Hadamard product}) \\
&= -\nu(z_l^{k+1} - z_l^k) - \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k) \\
&(0 \in \partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) + \nu(z_l^{k+1} - z_l^k))
\end{aligned}
$$

So

$$
\begin{aligned}
&\|\partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \| - \nu(z_l^{k+1} - z_l^k) - \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k)\| \\
&\leq \nu\|z_l^{k+1} - z_l^k\| + \nu\|\partial f_l(z_l^{k+1})\|\|q_l^{k+1} - q_l^k\| \\
&\text{(Cauthy-Schwartz Inequality and Triangle Inequality)} \\
&\leq \nu\|z_l^{k+1} - z_l^k\| + \nu M\|q_l^{k+1} - q_l^k\| (\|\partial f_l(z_l^{k+1})\| \leq M)
\end{aligned}
$$

For z_L^{k+1}, $\partial_{z_L^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) = 0$ by the optimality condition of Equation (4.4).

For q_l^{k+1},

$$\begin{aligned}
&\nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k}) + u_l^{k+1} - u_l^k \\
&= u_l^{k+1} - u_l^k \ (\nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k}) = 0 \\
&\text{by the optiamlity condition of Equation (4.5))}
\end{aligned}$$

So $\|\nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| = \|u_l^{k+1} - u_l^k\|$.
For u_l^{k+1},

$$\nabla_{u_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) = (p_{l+1}^{k+1} - q_l^{k+1}) = (u_l^{k+1} - u_l^k)/\rho$$

So $\|\nabla_{u_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| = \|u_l^{k+1} - u_l^k\|/\rho$.
In summary, we prove that $\nabla_{\mathrm{p}^{k+1}} L_\rho, \nabla_{\mathrm{W}^{k+1}} L_\rho, \partial_{\mathrm{z}^{k+1}} L_\rho, \nabla_{\mathrm{q}^{k+1}} L_\rho, \nabla_{\mathrm{u}^{k+1}} L_\rho$ are upper bounded by the linear combination of $\|\mathbf{p}^{k+1} - \mathbf{p}^k\|$,$\|\mathbf{W}^{k+1} - \mathbf{W}^k\|$, $\|\mathbf{z}^{k+1} - \mathbf{z}^k\|$, $\|\mathbf{q}^{k+1} - \mathbf{q}^k\|$, and $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|$. □

Proof of Theorem 8

Proof. To prove this theorem, we will first show that c_k satisfies two conditions: (1). $c_k \geq c_{k+1}$. (2). $\sum_{k=0}^{\infty} c_k$ is bounded. Specifically, first, we have

$$\begin{aligned}
c_k &= \min_{0\leq i\leq k}(\sum\nolimits_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1} - p_l^i\|_2^2 + \sum\nolimits_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1} - W_l^i\|_2^2 \\
&+ \sum\nolimits_{l=1}^{L-1} C_1\|z_l^{i+1} - z_l^i\|_2^2 + (\nu/2)\|z_L^{i+1} - z_L^i\|_2^2 + \sum\nolimits_{l=1}^{L-1} C_2\|q_l^{i+1} - q_l^i\|_2^2) \\
&\geq \min_{0\leq i\leq k+1}(\sum\nolimits_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1} - p_l^i\|_2^2 + \sum\nolimits_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1} - W_l^i\|_2^2 \\
&+ \sum\nolimits_{l=1}^{L-1} C_1\|z_l^{i+1} - z_l^i\|_2^2 + (\nu/2)\|z_L^{i+1} - z_L^i\|_2^2 + \sum\nolimits_{l=1}^{L-1} C_2\|q_l^{i+1} - q_l^i\|_2^2) \\
&= c_{k+1}
\end{aligned}$$

Therefore c_k satisfies the first condition. Second,

$$\begin{aligned}
&\sum_{k=0}^{\infty} c_k = \sum_{k=0}^{\infty} \min_{0\le i\le k}\Big(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1}-p_l^i\|_2^2 + \sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_1\|z_l^{i+1}-z_l^i\|_2^2 + (\nu/2)\|z_L^{i+1}-z_L^i\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{i+1}-q_l^i\|_2^2\Big) \\
&\le \sum_{k=0}^{\infty}\Big(\sum_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1}-p_l^k\|_2^2 + \sum_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1}-W_l^k\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_1\|z_l^{k+1}-z_l^k\|_2^2 + (\nu/2)\|z_L^{k+1}-z_L^k\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{k+1}-q_l^k\|_2^2\Big) \\
&\le L_\rho(\mathbf{p}^0,\mathbf{W}^0,\mathbf{z}^0,\mathbf{q}^0,\mathbf{u}^0) - L_\rho(\mathbf{p}^*,\mathbf{W}^*,\mathbf{z}^*,\mathbf{q}^*,\mathbf{u}^*)
\end{aligned}$$

(Lemma 5)

So c_k satisfies the second condition. Finally, since we have proved the first two conditions and the third one $c_k \ge 0$ is obvious, the convergence rate of $o(1/k)$ is proven (Lemma 1.2 in [26]). □

Appendix D

Appendix of the pdADMM-G Algorithm

D.1 Convergence Proofs

D.1.1 Preliminary Results

Lemma 24. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$u_l^k = \nu(q_l^k - f_l(z_l^k)).$$

Proof. This follows directly from the optimality condition of q_l^k and Equation (4.6). □

Lemma 25. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$\|u_l^{k+1} - u_l^k\| \leq \nu\|q_l^{k+1} - q_l^k\| + \nu S\|z_l^{k+1} - z_l^k\|.$$

Proof.

$$\begin{aligned}
&\|u_l^{k+1} - u_l^k\| \\
&= \|\nu(q_l^{k+1} - f_l(z_l^{k+1})) - \nu(q_l^k - f_l(z_l^k))\|(\text{Lemma 24}) \\
&\leq \nu\|q_l^{k+1} - q_l^k\| + \nu\|f_l(z_l^{k+1}) - f_l(z_l^k)\|(\text{Triangle Inequality}) \\
&\leq \nu\|q_l^{k+1} - q_l^k\| + \nu S\|z_l^{k+1} - z_l^k\|(\text{Assumption 6}).
\end{aligned}$$

□

Lemma 26. It holds for every $k \in \mathbb{N}$ and $l = 1, \cdots, L-1$ that

$$\|u_l^{k+1} - u_l^k\|_2^2 \leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + S^2\|z_l^{k+1} - z_l^k\|_2^2).$$

Proof.

$$\begin{aligned}
\|u_l^{k+1} - u_l^k\|_2^2 &= \nu^2\|q_l^{k+1} - f_l(z_l^{k+1}) - q_l^k + f_l(z_l^k)\|_2^2(\text{Lemma 24}) \\
&\leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + \|f_l(z_l^{k+1}) - f_l(z_l^k)\|_2^2)(\text{Mean Inequality}) \\
&\leq 2\nu^2(\|q_l^{k+1} - q_l^k\|_2^2 + S^2\|z_l^{k+1} - z_l^k\|_2^2)(\text{Assumption 6}).
\end{aligned}$$

□

Lemma 27. For every $k \in \mathbb{N}$, it holds that

$$L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \geq \sum_{l=2}^{L} (\tau_l^{k+1}/2) \|p_l^{k+1} - p_l^k\|_2^2, \tag{D.1}$$

$$L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \geq \sum_{l=1}^{L} (\theta_l^{k+1}/2) \|W_l^{k+1} - W_l^k\|_2^2, \tag{D.2}$$

$$L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) \geq (\nu/2) \sum_{l=1}^{L} \|z_l^{k+1} - z_l^k\|_2^2, \tag{D.3}$$

$$\beta_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \geq \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k), \tag{D.4}$$

$$\beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \geq \sum_{l=1}^{L} (\theta_l^{k+1}/2) \|W_l^{k+1} - W_l^k\|_2^2, \tag{D.5}$$

$$\beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) \geq (\nu/2) \sum_{l=1}^{L} \|z_l^{k+1} - z_l^k\|_2^2. \tag{D.6}$$

Proof. Generally, all inequalities can be obtained by applying optimality conditions of updating $\mathbf{p}$, $\mathbf{W}$, and $\mathbf{z}$, respectively. We only prove Inequalities (D.1), (D.3) and (D.4). This is because Inequalities (D.2) and (D.5) follow the same routine of Inequality (D.1), and Inequality (D.6) follows the same routine of Inequality (D.3).

Firstly, we focus on Inequality (D.1). The choice of τ_l^{k+1} requires

$$\phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) \leq U_l(p_l^{k+1}; \tau_l^{k+1}). \tag{D.7}$$

Moreover, the optimality condition of Equation (4.1) leads to

$$\nabla_{p_l^k} \phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + \tau_l^{k+1}(p_l^{k+1} - p_l^k) = 0. \tag{D.8}$$

Therefore

$$
\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&= \sum_{l=2}^{L} (\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) - \phi(p_l^{k+1}, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)) \\
&\geq \sum_{l=2}^{L} (\phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) - U_l(p_l^{k+1}; \tau_l^{k+1})) \text{(Inequality (D.7))} \\
&= \sum_{l=2}^{L} (-\nabla_{p_l^k} \phi^T(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k)(p_l^{k+1} - p_l^k) - (\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2) \\
&= \sum_{l=2}^{L} (\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 \text{(Equation (D.8))}.
\end{aligned}
$$

Next we prove Inequality (D.3). Because z_l^{k+1} minimizes Equation (4.3) and Equation (4.4), we have

$$
\begin{aligned}
&(\nu/2)\|z_l^{k+1} - W_l^{k+1} p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^{k+1})\|_2^2 + (\nu/2)\|z_l^{k+1} - z_l^k\|_2^2 \\
&\leq (\nu/2)\|z_l^k - W_l^{k+1} p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^k)\|_2^2, \qquad \text{(D.9)}
\end{aligned}
$$

and

$$
\begin{aligned}
&R(z_L^k; y) + (\nu/2)\|z_L^k - W_L^{k+1} p_L^{k+1}\|_2^2 - R(z_L^{k+1}; y) - (\nu/2)\|z_L^{k+1} - W_L^{k+1} p_L^{k+1}\|_2^2 \\
&= R(z_L^k; y) - R(z_L^{k+1}; y) + (\nu/2)\|z_L^k - z_L^{k+1}\|_2^2 + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1})^T (z_L^k - z_L^{k+1}) \\
&(\|a-b\|_2^2 - \|a-c\|_2^2 = \|b-c\|_2^2 + 2(c-a)^T(b-c) \text{ where } a = W_L^{k+1} p_L^{k+1},\ b = z_L^k, \\
&\text{and } c = z_L^{k+1}) \\
&\geq s^T(z_L^k - z_L^{k+1}) + (\nu/2)\|z_L^k - z_L^{k+1}\|_2^2 + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1})^T (z_L^k - z_L^{k+1}) \\
&(s \in \partial R(z_L^{k+1}; y) \text{ is a subgradient of } R(z_L^{k+1}; y)) \\
&= (\nu/2)\|z_L^{k+1} - z_L^k\|_2^2 \qquad \text{(D.10)} \\
&(0 \in s + \nu(z_L^{k+1} - W_L^{k+1} p_L^{k+1}) \text{ by the optimality condition of Equation (4.4)}).
\end{aligned}
$$

Therefore

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) \\
&= \sum\nolimits_{i=1}^{L-1}((\nu/2)\|z_l^k - W_l^{k+1}p_l^{k+1}\|_2^2 + (\nu/2)\|q_l^k - f_l(z_l^k)\|_2^2 \\
&- (\nu/2)\|z_l^{k+1} - W_l^{k+1}p_l^{k+1}\|_2^2 - (\nu/2)\|q_l^k - f_l(z_l^{k+1})\|_2^2) \\
&+ R(z_L^k; y) + (\nu/2)\|z_L^k - W_L^{k+1}p_L^{k+1}\|_2^2 - R(z_L^{k+1}; y) - (\nu/2)\|z_L^{k+1} - W_L^{k+1}p_L^{k+1}\|_2^2 \\
&\geq (\nu/2)\sum\nolimits_{l=1}^{L}\|z_l^{k+1} - z_l^k\|_2^2 (\text{Inequalities (D.9) and (D.10)}).
\end{aligned}
$$

Finally Inequality (D.4) follows directly the optimality condition of $\mathbf{p}^{k+1}$. □

Lemma 28. For every $k \in N$, it holds that

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \sum\nolimits_{l=1}^{L-1}((\rho/2 - 2\nu^2/\rho - \nu/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2) \qquad \text{(D.11)} \\
&\beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) - \beta_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \sum\nolimits_{l=1}^{L-1}((\rho/2 - 2\nu^2/\rho - \nu/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2). \qquad \text{(D.12)}
\end{aligned}
$$

Proof. We only prove Inequality (D.11) because Inequality (D.12) follows the same

routine of Inequality (D.11).

$$
\begin{aligned}
&L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) - L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \sum_{l=1}^{L-1}((\nu/2)\|f_l(z_l^{k+1}) - q_l^k\|_2^2 - (\nu/2)\|f_l(z_l^{k+1}) - q_l^{k+1}\|_2^2 - (u_l^{k+1})^T(q_l^k - q_l^{k+1}) \\
&+ (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&= \sum_{l=1}^{L-1}((\nu/2)\|f_l(z_l^{k+1}) - q_l^k\|_2^2 - (\nu/2)\|f_l(z_l^{k+1}) - q_l^{k+1}\|_2^2 \\
&- \nu(q_l^{k+1} - f_l(z_l^{k+1}))^T(q_l^k - q_l^{k+1}) + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&\text{(Lemma 24)} \\
&\geq \sum_{l=1}^{L-1}(-(\nu/2)\|q_l^{k+1} - q_l^k\|_2^2 + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (1/\rho)\|u_l^{k+1} - u_l^k\|_2^2) \\
&(-\nu(q_l - f_l(z_l^{k+1})) = -(\nu/2)\nabla_{q_l}\|q_l - f_l(z_l^{k+1})\|_2^2 \text{ is lipschitz continuous concerning } q_l \\
&\text{and Lemma 2.1 in [2])} \\
&\geq \sum_{l=1}^{L-1}(-(\nu/2)\|q_l^{k+1} - q_l^k\|_2^2 + (\rho/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2/\rho)\|q_l^{k+1} - q_l^k\|_2^2 \\
&- (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2) \quad \text{(Lemma 26)} \\
&= \sum_{l=1}^{L-1}((\rho/2 - 2\nu^2/\rho - \nu/2)\|q_l^{k+1} - q_l^k\|_2^2 - (2\nu^2 S^2/\rho)\|z_l^{k+1} - z_l^k\|_2^2).
\end{aligned}
$$

□

D.1.2 Proof of Lemma 8

Proof. We sum up Inequalities (D.1), (D.2), (D.3), and (D.11) to obtain Inequality (5.2), and we sum up Inequalities (D.4), (D.5), (D.6), and (D.12) to obtain Inequality (5.3). □

D.1.3 Proof of Lemma 9

Proof. (1) There exists $\mathbf{q}'$ such that $p_{l+1}^k = q_l'$ and

$$F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}') \geq \min_{\mathbf{p},\mathbf{W},\mathbf{z},\mathbf{q}}\{F(\mathbf{p}, \mathbf{W}, \mathbf{z}, \mathbf{q}) | p_{l+1} = q_l\} > -\infty.$$

Therefore, we have

$$
\begin{aligned}
&L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) \\
&= F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k) + \sum\nolimits_{l=1}^{L} (u_l^k)^T (p_{l+1}^k - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2 \\
&= R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^k - f_l(z_l^k)\|_2^2) \\
&+ \sum\nolimits_{l=1}^{L-1} ((u_l^k)^T (p_{l+1}^k - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&= R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^k - f_l(z_l^k)\|_2^2) \\
&+ \sum\nolimits_{l=1}^{L-1} (\nu(q_l^k - f_l(z_l^k))^T (q_l^{'} - q_l^k) + (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&(p_{l+1}^k = q_l^{'} \text{ and Lemma 24}) \\
&\geq R(z_L^k; y) + (\nu/2)(\sum\nolimits_{l=1}^{L} \|z_l^k - W_l^k p_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} \|q_l^{'} - f_l(z_l^k)\|_2^2) \\
&- \sum\nolimits_{l=1}^{L-1} (\nu/2)\|q_l^{'} - q_l^k\|_2^2 + \sum\nolimits_{l=1}^{L-1} (\rho/2)\|p_{l+1}^k - q_l^k\|_2^2) \\
&(\nu(q_l - f_l(z_l^{k+1})) = (\nu/2)\nabla_{q_l}\|q_l - f_l(z_l^{k+1})\|_2^2 \text{ is lipschitz continuous concerning } q_l \\
&\text{and Lemma 2.1 in [2])} \\
&= F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'}) + ((\rho - \nu)/2)\|p_{l+1}^k - q_l^k\|_2^2 > -\infty.
\end{aligned}
$$

Therefore, $F(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^{'})$ and $((\rho - \nu)/2)\|p_{l+1}^k - q_l^k\|_2^2$ are upper bounded by $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ and hence $L_\rho(\mathbf{p}^0, \mathbf{W}^0, \mathbf{z}^0, \mathbf{q}^0, \mathbf{u}^0)$ (Lemma 8). From Assumption 6, $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k)$ is bounded. $\mathbf{q}^k$ is also bounded because $(\rho - \nu)/2\|p_{l+1}^k - q_l^k\|_2^2$ is upper bounded. $\mathbf{u}^k$ is bounded because of Lemma 24.

(2). It follows the same routine as (1). □

D.1.4 Proof of Theorem 9

Proof. (1). From Lemmas 8 and 9, we know that $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent because a monotone bounded sequence converges. Moreover, we take the limit on

both sides of Inequality (5.2) to obtain

$$
\begin{aligned}
&0 = \lim_{k\to\infty} L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k) - \lim_{k\to\infty} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&\geq \lim_{k\to\infty}(\sum_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1} - p_l^k\|_2^2 + \sum_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1} - W_l^k\|_2^2 \\
&+ \sum_{l=1}^{L-1} C_1\|z_l^{k+1} - z_l^k\|_2^2 + (\nu/2)\|z_L^{k+1} - z_L^k\|_2^2 + \sum_{l=1}^{L-1} C_2\|q_l^{k+1} - q_l^k\|_2^2) \geq 0.
\end{aligned}
$$

Because $L_\rho(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ is convergent, then $\lim_{k\to\infty} \|\mathbf{p}^{k+1} - \mathbf{p}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_2^2 = 0$, $\lim_{k\to\infty} \|\mathbf{z}^{k+1} - \mathbf{z}^k\|_2^2 = 0$, and $\lim_{k\to\infty} \|\mathbf{q}^{k+1} - \mathbf{q}^k\|_2^2 = 0$. $\lim_{k\to\infty} \|\mathbf{u}^{k+1} - \mathbf{u}^k\|_2^2 = 0$ is derived from Lemma 26 in Section D.1 in the Appendix. (2). The proof follows the same procedure as (1). □

Proof of Lemma 10

Proof. (1). We know that $\partial L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})$
$= \{\nabla_{\mathrm{p}^{k+1}} L_\rho, \nabla_{\mathrm{W}^{k+1}} L_\rho, \partial_{\mathrm{z}^{k+1}} L_\rho, \nabla_{\mathrm{q}^{k+1}} L_\rho, \nabla_{\mathrm{u}^{k+1}} L_\rho\}$ [101]. Specifically, we prove that $\|g\|$ is upper bounded by the linear combination of $\|\mathbf{p}^{k+1} - \mathbf{p}^k\|$,$\|\mathbf{W}^{k+1} - \mathbf{W}^k\|$, $\|\mathbf{z}^{k+1} - \mathbf{z}^k\|$, $\|\mathbf{q}^{k+1} - \mathbf{q}^k\|$, and $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|$.
For p_l^{k+1},

$$
\begin{aligned}
&\nabla_{p_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{p_l^{k+1}} \phi(p_l^{k+1}, W_l^{k+1}, z_l^{k+1}, q_{l-1}^{k+1}, u_{l-1}^{k+1}) \\
&= \nabla_{p_l^k} \phi(p_l^k, W_l^k, z_l^k, q_{l-1}^k, u_{l-1}^k) + \tau_l^{k+1}(p_l^{k+1} - p_l^k) - \tau_l^{k+1}(p_l^{k+1} - p_l^k) \\
&+ \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k - \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k \\
&+ (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k) \\
&= -\tau_l^{k+1}(p_l^{k+1} - p_l^k) + \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k \\
&- \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k + (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k)
\end{aligned}
$$

(The optimality condition of Equation (4.1))

So

$$
\begin{aligned}
&\|\nabla_{p_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \| - \tau_l^{k+1}(p_l^{k+1} - p_l^k) + \nu(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - \nu(W_l^k)^T W_l^k p_l^k \\
&- \nu(W_l^{k+1})^T z_l^{k+1} + \nu(W_l^k)^T z_l^k + (u_{l-1}^{k+1} - u_{l-1}^k) + \rho(p_l^{k+1} - p_l^k) - \rho(q_{l-1}^{k+1} - q_{l-1}^k)\| \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|(W_l^{k+1})^T W_l^{k+1} p_l^{k+1} - (W_l^k)^T W_l^k p_l^k\| \\
&+ \nu\|(W_l^{k+1})^T z_l^{k+1} - (W_l^k)^T z_l^k\| + \|u_{l-1}^{k+1} - u_{l-1}^k\| + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\text{(Triangle Inequality)} \\
&= \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|(W_l^{k+1})^T W_l^{k+1}(p_l^{k+1} - p_l^k) + (W_l^{k+1})^T(W_l^{k+1} - W_l^k)p_l^k \\
&+ (W_l^{k+1} - W_l^k)^T W_l^k p_l^k\| + \nu\|(W_l^{k+1})^T(z_l^{k+1} - z_l^k) + (W_l^{k+1} - W_l^k)^T z_l^k\| \\
&+ \|u_{l-1}^{k+1} - u_{l-1}^k\| + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\|W_l^{k+1}\|^2\|p_l^{k+1} - p_l^k\| + \nu\|W_l^{k+1}\|\|W_l^{k+1} - W_l^k\|\|p_l^k\| \\
&+ \nu\|W_l^{k+1} - W_l^k\|\|W_l^k\|\|p_l^k\| + \nu\|W_l^{k+1}\|\|z_l^{k+1} - z_l^k\| + \nu\|W_l^{k+1} - W_l^k\|\|z_l^k\| \\
&+ \nu(\|q_{l-1}^{k+1} - q_{l-1}^k\| + S\|z_{l-1}^{k+1} - z_{l-1}^k\|) + \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \\
&\text{(Triangle Inequality, Cauthy-Schwartz Inequality and Lemma 20)} \\
&\leq \tau_l^{k+1}\|p_l^{k+1} - p_l^k\| + \nu\mathbb{N}_{\mathbf{W}}^2\|p_l^{k+1} - p_l^k\| + 2\nu\mathbb{N}_{\mathbf{W}}\mathbb{N}_{\mathbf{p}}\|W_l^{k+1} - W_l^k\| \\
&+ \nu\mathbb{N}_{\mathbf{W}}\|z_l^{k+1} - z_l^k\| + \nu\mathbb{N}_{\mathbf{z}}\|W_l^{k+1} - W_l^k\| + 2\nu^2(\|q_{l-1}^{k+1} - q_{l-1}^k\|_2^2 + S^2\|z_{l-1}^{k+1} - z_{l-1}^k\|_2^2) \\
&+ \rho\|p_l^{k+1} - p_l^k\| + \rho\|q_{l-1}^{k+1} - q_{l-1}^k\| \quad \text{(Lemma 6)}.
\end{aligned}
$$

For W_1^{k+1},

$$\begin{aligned}
&\nabla_{W_1^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{W_1^{k+1}} \phi(p_1^{k+1}, W_1^{k+1}, z_1^{k+1}) \\
&= \nabla_{W_1^k} \phi(p_1^{k+1}, W_1^k, z_1^k) + \theta_1^{k+1}(W_1^{k+1} - W_1^k) + \nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T \\
&- \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k) \\
&= \nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T - \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k)
\end{aligned}$$

(The optimality condition of Equation (4.2)).

So

$$\begin{aligned}
&\|\nabla_{W_1^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \|\nu(W_1^{k+1} - W_1^k) p_1^{k+1}(p_1^{k+1})^T - \nu(z_1^{k+1} - z_1^k)(p_1^{k+1})^T - \theta_1^{k+1}(W_1^{k+1} - W_1^k)\| \\
&\leq \nu\|W_1^{k+1} - W_1^k\|\|p_1^{k+1}\|^2 + \nu\|z_1^{k+1} - z_1^k\|\|p_1^{k+1}\| + \theta_1^{k+1}\|W_1^{k+1} - W_1^k\|
\end{aligned}$$

(Triangle Inequality and Cauthy-Schwartz Inequality)

$$\leq \nu\|W_1^{k+1} - W_1^k\|\mathbb{N}_{\mathbf{p}}^2 + \nu\|z_1^{k+1} - z_1^k\|\mathbb{N}_{\mathbf{p}} + \theta_1^{k+1}\|W_1^{k+1} - W_1^k\| \text{ (Theorem 6).}$$

For $W_l^{k+1} (1 < l \leq L)$,

$$\begin{aligned}
&\nabla_{W_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{W_l^{k+1}} \phi(p_l^{k+1}, W_l^{k+1}, z_l^{k+1}, p_{l-1}^{k+1}, u_{l-1}^{k+1}) \\
&= \nabla_{W_l^k} \phi(p_l^{k+1}, W_l^k, z_l^k, p_{l-1}^k, u_{l-1}^k) + \theta_l^{k+1}(W_l^{k+1} - W_1^k) + \nu(W_l^{k+1} - W_l^k) p_l^{k+1}(p_l^{k+1})^T \\
&- \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k) \\
&= \nu(W_l^{k+1} - W_l^k) p_l^{k+1}(p_l^{k+1})^T - \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k)
\end{aligned}$$

(The optimality condition of Equation (4.2)).

So

$$\begin{aligned}
&\|\nabla_{W_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \|\nu(W_l^{k+1} - W_l^k)p_l^{k+1}(p_l^{k+1})^T - \nu(z_l^{k+1} - z_l^k)(p_l^{k+1})^T - \theta_l^{k+1}(W_l^{k+1} - W_l^k)\| \\
&\leq \nu\|W_l^{k+1} - W_l^k\|\|p_l^{k+1}\|^2 + \nu\|z_l^{k+1} - z_l^k\|\|p_l^{k+1}\| + \theta_l^{k+1}\|W_l^{k+1} - W_l^k\| \\
&\text{(Triangle Inequality and Cauthy-Schwartz Inequality)} \\
&\leq \nu\|W_l^{k+1} - W_l^k\|\mathbb{N}_{\mathbf{p}}^2 + \nu\|z_l^{k+1} - z_l^k\|\mathbb{N}_{\mathbf{p}} + \theta_l^{k+1}\|W_l^{k+1} - W_l^k\| \quad \text{(Theorem 6)}.
\end{aligned}$$

For $z_l^{k+1}(l < L)$,

$$\begin{aligned}
&\partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) + \nu(z_l^{k+1} - z_l^k) - \nu(z_l^{k+1} - z_l^k) \\
&- \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k) \quad (\circ \text{ is Hadamard product}) \\
&= -\nu(z_l^{k+1} - z_l^k) - \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k) \\
&(0 \in \partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^k, \mathbf{u}^k) + \nu(z_l^{k+1} - z_l^k)).
\end{aligned}$$

So

$$\begin{aligned}
&\|\partial_{z_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| \\
&= \| - \nu(z_l^{k+1} - z_l^k) - \nu\partial f_l(z_l^{k+1}) \circ (q_l^{k+1} - q_l^k)\| \\
&\leq \nu\|z_l^{k+1} - z_l^k\| + \nu\|\partial f_l(z_l^{k+1})\|\|q_l^{k+1} - q_l^k\| \\
&\text{(Cauchy-Schwartz Inequality and Triangle Inequality)} \\
&\leq \nu\|z_l^{k+1} - z_l^k\| + \nu M\|q_l^{k+1} - q_l^k\| (\|\partial f_l(z_l^{k+1})\| \leq M).
\end{aligned}$$

For z_L^{k+1}, $\partial_{z_L^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) = 0$ by the optimality condition of Equation (4.4).

For q_l^{k+1},

$$\begin{aligned}
&\nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) \\
&= \nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k}) + u_l^{k+1} - u_l^k \\
&= u_l^{k+1} - u_l^k \text{ (by the optimality condition of Equation (4.5)).}
\end{aligned}$$

So $\|\nabla_{q_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| = \|u_l^{k+1} - u_l^k\|$.
For u_l^{k+1},

$$\nabla_{u_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1}) = (p_{l+1}^{k+1} - q_l^{k+1}) = (u_l^{k+1} - u_l^k)/\rho.$$

So $\|\nabla_{u_l^{k+1}} L_\rho(\mathbf{p}^{k+1}, \mathbf{W}^{k+1}, \mathbf{z}^{k+1}, \mathbf{q}^{k+1}, \mathbf{u}^{k+1})\| = \|u_l^{k+1} - u_l^k\|/\rho$.
In summary, we prove that $\nabla_{\mathrm{p}^{k+1}} L_\rho, \nabla_{\mathrm{W}^{k+1}} L_\rho, \partial_{\mathrm{z}^{k+1}} L_\rho, \nabla_{\mathrm{q}^{k+1}} L_\rho, \nabla_{\mathrm{u}^{k+1}} L_\rho$ are upper bounded by the linear combination of $\|\mathbf{p}^{k+1} - \mathbf{p}^k\|$,$\|\mathbf{W}^{k+1} - \mathbf{W}^k\|$, $\|\mathbf{z}^{k+1} - \mathbf{z}^k\|$, $\|\mathbf{q}^{k+1} - \mathbf{q}^k\|$, and $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|$.
(2). It follows exactly the proof of (1) except for p_l^{k+1} . □

D.1.5 The proof of Theorem 10

Proof. From Lemma 9(1), $(\mathbf{p}^k, \mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ because a bounded sequence has at least a limit point. From Lemma 10 and Theorem 9, $\|g^{k+1}\| \to 0$ as $k \to \infty$. According to the definition of general subgradient (Definition 8.3 in [84]), we have $0 \in \partial L_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$. In other words, every limit point $(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ is a stationary point. □

D.1.6 The proof of Theorem 11

Proof. From Lemma 9(2), $(\mathbf{W}^k, \mathbf{z}^k, \mathbf{q}^k, \mathbf{u}^k)$ has at least a limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$ because a bounded sequence has at least a limit point. $\mathbf{p}^k$ has at least a limit point

$\mathbf{p}^*$ because $\mathbf{p}^k \in \Delta$ and Δ is finite. From Lemma 10(2) and Theorem 9, $\|\overline{g}_{\mathbf{W}}^{k+1}\| \to 0$, $\|\overline{g}_{\mathbf{z}}^{k+1}\| \to 0$, $\|\overline{g}_{\mathbf{q}}^{k+1}\| \to 0$, $\|\overline{g}_{\mathbf{u}}^{k+1}\| \to 0$ as $k \to \infty$. According to the definition of general subgradient (Defintion 8.3 in [84]), we have $\nabla_{\mathbf{W}^*}\beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0$, $0 \in \partial_{\mathbf{z}^*}\beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*)$, $\nabla_{\mathbf{q}^*}\beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0$ and $\nabla_{\mathbf{u}^*}\beta_\rho(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*) = 0$ (i.e. $p_{l+1}^* = q_l^*$). In other words, every limit point $(\mathbf{W}^*, \mathbf{z}^*, \mathbf{u}^*)$ is a stationary point of Problem 7. Moreover, τ_l^k has a limit point τ_l^* because it is bounded. Let $\boldsymbol{\tau}^k = \{\tau_l^k\}_{l=2}^L$. Consider a subsequence $(\mathbf{p}^s, \mathbf{W}^s, \mathbf{z}^s, \mathbf{q}^s, \mathbf{u}^s, \boldsymbol{\tau}^{s+1}) \to (\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*, \mathbf{u}^*, \boldsymbol{\tau}^*)$. Because $u_l^{s+1} = u_l^s + \rho(p_{l+1}^s - q_l^s)$ and $u_l^{s+1} \to u_l^s$, thus $p_{l+1}^s \to q_l^s$, and $p_{l+1}^{s+1} \to q_l^{s+1}$. Because $q_l^{s+1} \to q_l^s$, then $p_{l+1}^{s+1} \to p_{l+1}^s$ for any l. In other words, $\mathbf{p}^{s+1} \to \mathbf{p}^s$. Because $\mathbf{p}^s \to \mathbf{p}^*$, then $\mathbf{p}^{s+1} \to \mathbf{p}^*$. The optimality condition of $\mathbf{p}^{s+1}$ (i.e. Equation (5.1)) leads to

$$p_l^{s+1} \leftarrow \arg\min_{\delta \in \Delta} \|\delta - p_l^s - \nabla_{p_l^s}\phi(p_l^s, W_l^s, z_l^s, q_{l-1}^s, u_{l-1}^s)/\tau_l^{s+1})\|.$$

Taking $s \to \infty$ on both sides, we have

$$p_l^* \leftarrow \arg\min_{\delta \in \Delta} \|\delta - (p_l^* - \nabla_{p_l^*}\phi(p_l^*, W_l^*, z_l^*, q_{l-1}^*, u_{l-1}^*)/\tau_l^*)\|.$$

Because $\nabla_{p_l^*}F(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*) = \nu W_l^T(z_l^* - W_l^* p_l^*) = \nabla_{p_l^*}\phi(p_l^*, W_l^*, z_l^*, q_{l-1}^*, u_{l-1}^*)$. Then

$$p_l^* \leftarrow \arg\min_{\delta \in \Delta} \|\delta - (p_l^* - \nabla_{p_l^*}F(\mathbf{p}^*, \mathbf{W}^*, \mathbf{z}^*, \mathbf{q}^*)/\tau_l^*)\|.$$

Namely, $\mathbf{p}^*$ is a quantized stationary point of Problem 7. □

D.1.7 The proof of Theorem 12

Proof. (1). To prove this, we will first show that c_k satisfies two conditions: (1). $c_k \geq c_{k+1}$. (2). $\sum_{k=0}^{\infty} c_k$ is bounded. We then conclude the convergence rate of $o(1/k)$

based on these two conditions. Specifically, first, we have

$$\begin{aligned}
c_k =& \min_{0\le i\le k}\Big(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1}-p_l^i\|_2^2+\sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2\\
&+\sum_{l=1}^{L-1}C_1\|z_l^{i+1}-z_l^i\|_2^2+(\nu/2)\|z_L^{i+1}-z_L^i\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{i+1}-q_l^i\|_2^2\Big)\\
\ge& \min_{0\le i\le k+1}\Big(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1}-p_l^i\|_2^2+\sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2\\
&+\sum_{l=1}^{L-1}C_1\|z_l^{i+1}-z_l^i\|_2^2+(\nu/2)\|z_L^{i+1}-z_L^i\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{i+1}-q_l^i\|_2^2\Big)\\
=& \, c_{k+1}.
\end{aligned}$$

Therefore c_k satisfies the first condition. Second,

$$\begin{aligned}
&\sum_{k=0}^{\infty}c_k\\
&=\sum_{k=0}^{\infty}\min_{0\le i\le k}\Big(\sum_{l=2}^{L}(\tau_l^{i+1}/2)\|p_l^{i+1}-p_l^i\|_2^2+\sum_{l=1}^{L}(\theta_l^{i+1}/2)\|W_l^{i+1}-W_l^i\|_2^2\\
&+\sum_{l=1}^{L-1}C_1\|z_l^{i+1}-z_l^i\|_2^2+(\nu/2)\|z_L^{i+1}-z_L^i\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{i+1}-q_l^i\|_2^2\Big)\\
&\le\sum_{k=0}^{\infty}\Big(\sum_{l=2}^{L}(\tau_l^{k+1}/2)\|p_l^{k+1}-p_l^k\|_2^2+\sum_{l=1}^{L}(\theta_l^{k+1}/2)\|W_l^{k+1}-W_l^k\|_2^2\\
&+\sum_{l=1}^{L-1}C_1\|z_l^{k+1}-z_l^k\|_2^2+(\nu/2)\|z_L^{k+1}-z_L^k\|_2^2+\sum_{l=1}^{L-1}C_2\|q_l^{k+1}-q_l^k\|_2^2\Big)\\
&\le L_\rho(\mathbf{p}^0,\mathbf{W}^0,\mathbf{z}^0,\mathbf{q}^0,\mathbf{u}^0)-L_\rho(\mathbf{p}^*,\mathbf{W}^*,\mathbf{z}^*,\mathbf{q}^*,\mathbf{u}^*)(\text{Lemma 8}).
\end{aligned}$$

So $\sum_{k=0}^{\infty}c_k$ is bounded and c_k satisfies the second condition. Finally, it has been proved that the sufficient conditions of convergence rate $o(1/k)$ are: (1) $c_k \ge c_{k+1}$, and (2) $\sum_{k=0}^{\infty}c_k$ is bounded, and (3) $c_k \ge 0$ (Lemma 1.2 in [26]). Since we have proved the first two conditions and the third one $c_k \ge 0$ is obvious, the convergence rate of $o(1/k)$ is proven.

(2). It follows the same procedure as (1). □

D.2 More Experimental Results

D.2.1 The Settings of All Hyperparameters

This section provides more details on the hyperparameter settings of all datasets, lr denotes learning rate, which are shown in the following tables.

Dataset	Cora	PubMed	Citeseer
lr(GD)	10^{-1}	5×10^{-2}	10^{-1}
lr(Adadelta)	10^{-3}	10^{-3}	10^{-3}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-4}	10^{-4}	10^{-3}
ρ, ν(pdADMM-G)	10^{-4}	10^{-4}	10^{-4}
ρ, ν(pdADMM-G-Q)	10^{-4}	10^{-3}	10^{-3}

Dataset	Amazon Computers	Amazon Photo	Coauthor CS
lr(GD)	10^{-2}	10^{-2}	10^{-1}
lr(Adadelta)	10^{-3}	10^{-3}	10^{-3}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-3}	10^{-3}	10^{-3}
ρ, ν(pdADMM-G)	10^{-3}	10^{-3}	10^{-2}
ρ, ν(pdADMM-G-Q)	10^{-3}	10^{-3}	10^{-2}

Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
lr(GD)	10^{-1}	10^{-3}	10^{-2}
lr(Adadelta)	10^{-3}	10^{-2}	10^{-1}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-3}	10^{-3}	10^{-3}
ρ, ν(pdADMM-G)	10^{-2}	10^{-4}	10^{-4}
ρ, ν(pdADMM-G-Q)	10^{-2}	10^{-4}	10^{-4}

Table D.1: Hyperparameter settings of all methods on nine benchmark datasets when the number of neurons is 100.

Dataset	Cora	PubMed	Citeseer
lr(GD)	10^{-1}	5×10^{-3}	10^{-1}
lr(Adadelta)	10^{-3}	10^{-4}	10^{-3}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-4}	10^{-4}	10^{-4}
ρ, ν(pdADMM-G)	10^{-4}	10^{-4}	10^{-3}
ρ, ν(pdADMM-G-Q)	10^{-4}	10^{-3}	10^{-3}

Dataset	Amazon Computers	Amazon Photo	Coauthor CS
lr(GD)	10^{-2}	10^{-2}	10^{-1}
lr(Adadelta)	10^{-3}	10^{-3}	10^{-3}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-4}	10^{-4}	10^{-4}
ρ, ν(pdADMM-G)	10^{-3}	10^{-3}	10^{-3}
ρ, ν(pdADMM-G-Q)	10^{-3}	10^{-3}	10^{-3}

Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
lr(GD)	10^{-2}	10^{-2}	10^{-2}
lr(Adadelta)	10^{-3}	10^{-2}	10^{-1}
lr(Adagrad)	10^{-3}	10^{-3}	10^{-3}
lr(Adam)	10^{-4}	10^{-3}	10^{-3}
ρ, ν(pdADMM-G)	10^{-2}	10^{-4}	10^{-4}
ρ, ν(pdADMM-G-Q)	10^{-2}	10^{-4}	10^{-4}

Table D.2: Hyperparameter settings of all methods on nine benchmark datasets when the number of neurons is 500.

D.2.2 The Performance of Validation Sets

This section provides more experimental results on the validation sets of all datasets, which are shown in the following tables.

Dataset	Cora	PubMed	Citeseer
GD	0.704 ±0.037	0.626 ± 0.072	0.619 ±0.045
Adadelta	0.652 ±0.064	0.720 ±0.035	0.620 ±0.022
Adagrad	0.720 ± 0.022	0.762 ± 0.012	0.604 ±0.027
Adam	0.720 ±0.034	0.745 ± 0.014	0.624 ±0.014
pdADMM-G	0.750 ±0.005	0.788 ±0.004	**0.724 ±0.005**
pdADMM-G-Q	**0.754 ± 0.002**	**0.793 ±0.002**	0.722 ±0.002

Dataset	Amazon Computers	Amazon Photo	Coauthor CS
GD	0.654 ±0.033	0.730 ±0.165	0.875 ±0.007
Adadelta	0.136 ±0.062	0.343 ±0.046	0.781 ±0.084
Adagrad	0.750 ±0.095	0.808 ±0.018	0.889 ±0.006
Adam	0.740 ±0.010	**0.850 ±0.006**	0.887 ±0.009
pdADMM-G	**0.753 ±0.005**	0.846 ±0.014	0.913 ±0.003
pdADMM-G-Q	0.688 ±0.063	0.822 ±0.013	**0.916 ±0.003**

Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
GD	0.921 ±0.009	0.464 ±0.008	0.378 ±0.004
Adadelta	0.918 ±0.014	0.461 ±0.006	0.514 ±0.014
Adagrad	0.928 ±0.005	0.480 ±0.003	0.574 ±0.008
Adam	0.919 ± 0.010	0.512 ±0.004	**0.681 ±0.003**
pdADMM-G	0.933 ±0.001	**0.514 ±0.001**	0.649 ±0.012
pdADMM-G-Q	**0.935 ±0.002**	0.506 ±0.004	0.661 ±0.004

Table D.3: The validation performance of all methods when the number of neurons is 100.

Dataset	Cora	PubMed	Citeseer
GD	0.731 ±0.018	0.651 ±0.034	0.679 ±0.008
Adadelta	0.716 ±0.061	0.688 ±0.024	0.597 ±0.025
Adagrad	0.765 ±0.014	0.776 ±0.006	0.668 ±0.028
Adam	**0.758 ±0.013**	0.778 ±0.008	0.668 ±0.020
pdADMM-G	0.753 ±0.004	**0.792 ±0.004**	0.729 ±0.003
pdADMM-G-Q	0.757 ±0.005	**0.792 ±0.003**	**0.730 ±0.004**

Dataset	Amazon Computers	Amazon Photo	Coauthor CS
GD	0.727 ± 0.012	0.809 ±0.012	0.897 ±0.003
Adadelta	0.246 ±0.073	0.371 ±0.075	0.884 ±0.003
Adagrad	0.766 ±0.011	0.860 ±0.003	**0.912 ±0.004**
Adam	0.750 ±0.017	**0.872 ±0.020**	0.893 ±0.013
pdADMM-G	**0.778 ±0.007**	0.861 ±0.005	**0.912 ±0.003**
pdADMM-G-Q	0.764 ±0.008	0.850 ±0.009	0.910 ±0.003

Dataset	Coauthor Physics	Flickr	Ogbn-Arxiv
GD	0.928 ±0.001	0.466 ±0.001	0.451 ±0.033
Adadelta	0.932 ±0.006	0.462 ±0.004	0.591 ±0.017
Adagrad	**0.935 ±0.005**	0.488 ±0.007	0.646 ±0.010
Adam	0.933 ±0.007	**0.516 ±0.002**	**0.692 ±0.008**
pdADMM-G	0.932 ±0.001	0.514 ±0.003	0.661 ±0.005
pdADMM-G-Q	0.933 ±0.002	0.514 ±0.001	0.667 ±0.003

Table D.4: The validation performance of all methods when the number of neurons is 500.

Bibliography

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[2] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[3] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 2006.

[4] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[5] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

[6] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1): 1–122, 2011.

[7] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral

networks and locally connected networks on graphs. *International Conference on Learning Representations (ICLR)*, 2014.

[8] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Deep neural networks for learning graph representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[9] Miguel Carreira-Perpinan and Weiran Wang. Distributed optimization of deeply nested systems. In *Artificial Intelligence and Statistics*, pages 10–19, 2014.

[10] Tsung-Hui Chang. A proximal dual consensus admm method for multi-agent constrained optimization. *IEEE Transactions on Signal Processing*, 64(14): 3719–3734, 2016.

[11] Tsung-Hui Chang, Mingyi Hong, and Xiangfeng Wang. Multi-agent distributed optimization via inexact consensus admm. *IEEE Transactions on Signal Processing*, 63(2):482–497, 2014.

[12] Tsung-Hui Chang, Mingyi Hong, Wei-Cheng Liao, and Xiangfeng Wang. Asynchronous distributed admm for large-scale optimization—part i: Algorithm and convergence analysis. *IEEE Transactions on Signal Processing*, 64(12):3118–3130, 2016.

[13] Tsung-Hui Chang, Wei-Cheng Liao, Mingyi Hong, and Xiangfeng Wang. Asynchronous distributed admm for large-scale optimization—part ii: Linear convergence analysis and numerical performance. *IEEE Transactions on Signal Processing*, 64(12):3131–3144, 2016.

[14] Rick Chartrand and Brendt Wohlberg. A nonconvex admm algorithm for group sparsity with sparse groups. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6009–6013. IEEE, 2013.

[15] Lei Chen, Zhengdao Chen, and Joan Bruna. On graph neural networks versus graph-augmented mlps. In *Ninth International Conference on Learning Representations*, 2021.

[16] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015.

[17] François Chollet et al. Keras. https://keras.io, 2015.

[18] Francois Chollet. *Deep learning with python*. Manning Publications Co., 2017.

[19] Anna Choromanska, Benjamin Cowen, Sadhana Kumaravel, Ronny Luss, Mattia Rigotti, Irina Rish, Paolo Diachille, Viatcheslav Gurev, Brian Kingsbury, Ravi Tejwani, et al. Beyond backprop: Online alternating minimization with auxiliary variables. In *International Conference on Machine Learning*, pages 1193–1202. PMLR, 2019.

[20] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. In *NeurIPS 2018 Workshop on Machine Learning for Creativity and Design*, 2018.

[21] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2022.

[22] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pages 3123–3131, 2015.

[23] Hanjun Dai, Zornitsa Kozareva, Bo Dai, Alex Smola, and Le Song. Learning

steady-states of iterative algorithms over graphs. In *International conference on machine learning*, pages 1106–1114. PMLR, 2018.

[24] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

[25] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 3844–3852, 2016.

[26] Wei Deng, Ming-Jun Lai, Zhimin Peng, and Wotao Yin. Parallel multi-block admm with o (1/k) convergence. *Journal of Scientific Computing*, 71(2):712–736, 2017.

[27] Simon S Du, Chi Jin, Jason D Lee, Michael I Jordan, Aarti Singh, and Barnabas Poczos. Gradient descent can take exponential time to escape saddle points. *Advances in neural information processing systems*, 30, 2017.

[28] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[29] Claudio Gallicchio and Alessio Micheli. Graph echo state networks. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2010.

[30] Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2): 59–99, 2016.

[31] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[32] Stefanie Gunther, Lars Ruthotto, Jacob B Schroder, Eric C Cyr, and Nicolas R Gauger. Layer-parallel training of deep residual neural networks. *SIAM Journal on Mathematics of Data Science*, 2(1):1–23, 2020.

[33] Ke Guo, Deren Han, David ZW Wang, and Tingting Wu. Convergence of admm for multi-block nonconvex separable optimization models. *Frontiers of Mathematics in China*, 12(5):1139–1162, 2017.

[34] Davood Hajinezhad and Mingyi Hong. Nonconvex alternating direction method of multipliers for distributed sparse principal component analysis. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 255–259. IEEE, 2015.

[35] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035, 2017.

[36] Sayed Hadi Hashemi, Sangeetha Abdu Jyothi, and Roy H Campbell. Tictac: Accelerating distributed deep learning with communication scheduling. In *Proceedings of the 2nd SysML Conference*, 2019.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[38] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *preprint arXiv:1506.05163*, 2015.

[39] Mingyi Hong. A distributed, asynchronous, and incremental algorithm for nonconvex optimization: an admm approach. *IEEE Transactions on Control of Network Systems*, 5(3):935–945, 2017.

[40] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

[41] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in neural information processing systems*, pages 22118–22133, 2020.

[42] Tianjian Huang, Prajwal Singhania, Maziar Sanjabi, Pabitra Mitra, and Meisam Razaviyayn. Alternating direction method of multipliers for quantization. In *International Conference on Artificial Intelligence and Statistics*, pages 208–216. PMLR, 2021.

[43] Zhouyuan Huo, Bin Gu, and Heng Huang. Training neural networks using features replay. In *Advances in Neural Information Processing Systems*, pages 6659–6668, 2018.

[44] Zhouyuan Huo, Bin Gu, Heng Huang, et al. Decoupled parallel backpropagation with convergence guarantee. In *International Conference on Machine Learning*, pages 2098–2106. PMLR, 2018.

[45] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[46] Md Saiful Islam, Tonmoy Sarkar, Sazzad Hossain Khan, Abu-Hena Mostofa Kamal, SM Murshid Hasan, Alamgir Kabir, Dalia Yeasmin, Mohammad Ariful

Islam, Kamal Ibne Amin Chowdhury, Kazi Selim Anwar, et al. Covid-19–related infodemic and its impact on public health: A global social media analysis. *The American journal of tropical medicine and hygiene*, 103(4):1621, 2020.

[47] Gauri Jagatap and Chinmay Hegde. Learning relu networks via alternating minimization. *arXiv preprint arXiv:1806.07863*, 2018.

[48] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 5308–5317, 2016.

[49] Jae-wook Jang, Jiyoung Woo, Jaesung Yun, and Huy Kang Kim. Mal-netminer: malware classification based on social network analysis of call graph. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 731–734, 2014.

[50] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

[51] Farkhondeh Kiaee, Christian Gagné, and Mahdieh Abbasi. Alternating direction method of multipliers for sparse convolutional neural networks. *arXiv preprint arXiv:1611.01590*, 2016.

[52] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[53] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[55] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[56] Sandeep Kumar, Rahul Jain, and Ketan Rajawat. Asynchronous optimization over heterogeneous networks via consensus admm. *IEEE Transactions on Signal and Information Processing over Networks*, 3(1):114–129, 2016.

[57] Tim Tsz-Kit Lau, Jinshan Zeng, Baoyuan Wu, and Yuan Yao. A proximal block coordinate descent algorithm for deep neural network training. In *International Conference on Learning Representations Workshop*, 2018.

[58] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11):2278–2324, 1998.

[59] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436, 2015.

[60] Guoyin Li and Ting Kei Pong. Global convergence of splitting methods for non-convex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460, 2015.

[61] Hongyi Li, Junxiang Wang, Yongchao Wang, Yue Cheng, and Liang Zhao. Community-based layerwise distributed training of graph convolutional networks. *NeurIPS 2021 Workshop on Optimization for Machine Learning (OPT 2021)*, 2021.

[62] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

[63] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *International Conference on Learning Representations (ICLR)*, 2016.

[64] Qinghua Liu, Xinyue Shen, and Yuantao Gu. Linearized admmfor nonconvex nonsmooth optimization with convergence analysis. *IEEE Access*, 7:76131–76144, 2019.

[65] Liangchen Luo, Yuanhao Xiong, and Yan Liu. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*, 2019.

[66] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. Citeseer, 2013.

[67] Sindri Magnússon, Pradeep Chathuranga Weeraddana, Michael G Rabbat, and Carlo Fischione. On the convergence of alternating direction lagrangian methods for nonconvex structured optimization problems. *IEEE Transactions on Control of Network Systems*, 3(3):296–309, 2015.

[68] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

[69] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev

Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[70] João FC Mota, João MF Xavier, Pedro MQ Aguiar, and Markus Püschel. Distributed admm for model predictive control and congestion control. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5110–5115. IEEE, 2012.

[71] Geoffrey Negiar, Armin Askari, Fabian Pedregosa, and Laurent El Ghaoui. Lifted neural networks for weight initialization. *10th NIPS Workshop on Optimization for Machine Learning (NIPS 2017)*, 2017.

[72] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL `http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf`.

[73] Elizabeth Newman, Lars Ruthotto, Joseph Hart, and Bart van Bloemen Waanders. Train like a (var) pro: efficient training of neural networks with variable projection. *SIAM Journal on Mathematics of Data Science*, 3(4):1041–1066, 2021.

[74] Elizabeth Newman, Julianne Chung, Matthias Chung, and Lars Ruthotto. slimtrain—a stochastic approximation method for training separable deep neural networks. *SIAM Journal on Scientific Computing*, 44(4):A2322–A2348, 2022.

[75] Mark EJ Newman, Stephanie Forrest, and Justin Balthrop. Email networks and the spread of computer viruses. *Physical Review E*, 66(3):035101, 2002.

[76] Seol-Hyun Noh. Analysis of gradient vanishing of rnns and performance comparison. *Information*, 12(11):442, 2021.

[77] Beng Chin Ooi, Kian-Lee Tan, Sheng Wang, Wei Wang, Qingchao Cai, Gang Chen, Jinyang Gao, Zhaojing Luo, Anthony KH Tung, Yuan Wang, et al. Singa: A distributed deep learning platform. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 685–688. ACM, 2015.

[78] S Pan, R Hu, G Long, J Jiang, L Yao, and C Zhang. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI International Joint Conference on Artificial Intelligence*, 2018.

[79] Panos Parpas and Corey Muir. Predict globally, correct locally: Parallel-in-time optimal control of neural networks. *arXiv preprint arXiv:1902.02542*, 2019.

[80] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5): 1–17, 1964.

[81] Linbo Qiao, Tao Sun, Hengyue Pan, and Dongsheng Li. Inertial proximal deep learning alternating minimization for efficient neutral network training. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3895–3899. IEEE, 2021.

[82] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[83] Herbert Robbins and S Monro. A stochastic approximation method, annals math. *Statistics*, 22:400–407, 1951.

[84] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[85] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[86] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[87] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*, pages 362–373. Springer, 2018.

[88] Alexander Sergeev and Mike Del Balso. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799*, 2018.

[89] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop (R2L),NeurIPS*, 2018.

[90] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.

[91] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[92] Yu Tang, Zhigang Kan, Dequan Sun, Linbo Qiao, Jingjing Xiao, Zhiquan Lai, and Dongsheng Li. Admmirnn: Training rnn with stable convergence via an efficient admm approach. In Frank Hutter, Kristian Kersting, Jefrey Lijffijt, and Isabel Valera, editors, *Machine Learning and Knowledge Discovery in Databases*,

pages 3–18, Cham, 2021. Springer International Publishing. ISBN 978-3-030-67661-2.

[93] Gavin Taylor, Ryan Burmeister, Zheng Xu, Bharat Singh, Ankit Patel, and Tom Goldstein. Training neural networks without gradients: A scalable admmapproach. In *International Conference on Machine Learning*, pages 2722–2731, 2016.

[94] Andreas Themelis and Panagiotis Patrinos. Douglas–rachford splitting and admm for nonconvex optimization: Tight convergence results. *SIAM Journal on Optimization*, 30(1):149–181, 2020.

[95] T Tieleman and G Hinton. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. Technical report, Technical Report., 2017.

[96] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=7UmjRGzp-A`.

[97] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234, 2016.

[98] Junxiang Wang and Liang Zhao. The application of multi-block admm on isotonic regression problems. *11th Workshop on Optimization for Machine Learning (OPT 2019), co-located with NeurIPS 2019*, 2019.

[99] Junxiang Wang and Liang Zhao. Nonconvex generalization of alternating direction method of multipliers for nonlinear equality constrained problems. *Results*

in Control and Optimization, page 100009, 2021. ISSN 2666-7207. doi: https://doi.org/10.1016/j.rico.2021.100009. URL `https://www.sciencedirect.com/science/article/pii/S2666720721000035`.

[100] Junxiang Wang and Liang Zhao. Convergence and applications of alternating direction method of multipliers on the multi-convex problems. *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2022.

[101] Junxiang Wang, Fuxun Yu, Xiang Chen, and Liang Zhao. Admm for efficient deep learning with global convergence. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 111–119, 2019.

[102] Junxiang Wang, Zheng Chai, Yue Chen, and Liang Zhao. Tunable subnetwork splitting for model-parallelism of neural network training. In *ICML 2020 Workshop: Beyond First Order Methods in Machine Learning*, 2020.

[103] Junxiang Wang, Zheng Chai, Yue Cheng, and Liang Zhao. Toward model parallelism for deep neural network based on gradient-free admmframework. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 591–600. IEEE, 2020.

[104] Junxiang Wang, Junji Jiang, and Liang Zhao. An invertible graph diffusion neural network for source localization. In *Proceedings of the ACM Web Conference 2022*, pages 1058–1069, 2022.

[105] Junxiang Wang, Hongyi Li, Zheng Chai, Yongchao Wang, Yue Cheng, and Liang Zhao. Towards quantized model parallelism for graph-augmented mlps based on gradient-free admm framework. *IEEE Transactions on Neural Networks and Lerning Systems*, 2022.

[106] Junxiang Wang, Hongyi Li, and Liang Zhao. Accelerated gradient-free neural network training by multi-convex alternating optimization. *Neurocomputing*, 487:130–143, 2022.

[107] Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of admm in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, pages 1–35, 2015.

[108] Ermin Wei and Asuman Ozdaglar. Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5445–5450. IEEE, 2012.

[109] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. In *Advances in neural information processing systems*, pages 1509–1519, 2017.

[110] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019.

[111] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 2020.

[112] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[113] Xingyu Xie, Jianlong Wu, Guangcan Liu, Zhisheng Zhong, and Zhouchen Lin. Differentiable linearized admm. In *International Conference on Machine Learning*, pages 6902–6911, 2019.

[114] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[115] Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.

[116] Zheng Xu, Gavin Taylor, Hao Li, Mário AT Figueiredo, Xiaoming Yuan, and Tom Goldstein. Adaptive consensus admmfor distributed optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3841–3850. JMLR. org, 2017.

[117] Babak Zamanlooy and Mitra Mirhassani. Efficient vlsi implementation of neural networks with hyperbolic tangent activation function. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(1):39–48, 2014.

[118] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[119] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020.

[120] Jinshan Zeng, Tim Tsz-Kit Lau, Shaobo Lin, and Yuan Yao. Global convergence of block coordinate descent in deep learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7313–7323, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[121] Guoqiang Zhang and W Bastiaan Kleijn. Training deep neural networks via optimization over graphs. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4119–4123. IEEE, 2018.

[122] Hao Zhang, Zeyu Zheng, Shizhen Xu, Wei Dai, Qirong Ho, Xiaodan Liang, Zhiting Hu, Jinliang Wei, Pengtao Xie, and Eric P Xing. Poseidon: An efficient communication architecture for distributed deep learning on gpu clusters. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*, pages 181–193, 2017.

[123] Ruiliang Zhang and James Kwok. Asynchronous distributed admm for consensus optimization. In *International Conference on Machine Learning*, pages 1701–1709, 2014.

[124] Ziming Zhang and Matthew Brand. Convergent block coordinate descent for training tikhonov regularized deep neural networks. In *Advances in Neural Information Processing Systems*, pages 1721–1730, 2017.

[125] Ziming Zhang, Yuting Chen, and Venkatesh Saligrama. Efficient training of very deep neural networks for supervised hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1487–1495, 2016.

[126] Shengyu Zhu, Mingyi Hong, and Biao Chen. Quantized consensus admm for multi-agent distributed optimization. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4134–4138. IEEE, 2016.

[127] Huiping Zhuang, Yi Wang, Qinglai Liu, and Zhiping Lin. Fully decoupled neural network learning using delayed gradients. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[128] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. *Advances in Neural Information Processing Systems*, 33, 2020.

[129] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010.