8-29-2018

Study on efficient sparse and low-rank optimization and its applications

Jian Lou

Follow this and additional works at: https://repository.hkbu.edu.hk/etd_oa

Recommended Citation

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at HKBU Institutional Repository. It has been accepted for inclusion in Open Access Theses and Dissertations by an authorized administrator of HKBU Institutional Repository. For more information, please contact repository@hkbu.edu.hk.
DATE: August 29, 2018

STUDENT'S NAME: LOU Jian

THESIS TITLE: Study on Efficient Sparse and Low-rank Optimization and Its Applications

This is to certify that the above student's thesis has been examined by the following panel members and has received full approval for acceptance in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Chairman: Dr Tong Tiejun
Associate Professor, Department of Mathematics, HKBU
(Designated by Dean of Faculty of Science)

Internal Members: Dr Tam Hon Wah
Associate Professor, Department of Computer Science, HKBU
(Designated by Head of Department of Computer Science)

Dr Huang Xin
Assistant Professor, Department of Computer Science, HKBU

External Members: Prof Ji Ping
Professor
Department of Industrial and Systems Engineering
The Hong Kong Polytechnic University

Prof Tan Kay Chen
Professor
Department of Computer Science
City University of Hong Kong

In-attendance: Prof Cheung Yiu Ming
Professor, Department of Computer Science, HKBU

Issued by Graduate School, HKBU
Study on Efficient Sparse and Low-rank Optimization and Its Applications

LOU Jian

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Principal Supervisor: Prof. CHEUNG Yiu-ming

Hong Kong Baptist University

August 2018
DECLARATION

I hereby declare that this thesis represents my own work which has been done after registration for the degree of PhD at Hong Kong Baptist University, and has not been previously included in a thesis or dissertation submitted to this or any other institution for a degree, diploma or other qualifications.

I have read the University’s current research ethics guidelines, and accept responsibility for the conduct of the procedures in accordance with the University’s Committee on the Use of Human & Animal Subjects in Teaching and Research (HASC). I have attempted to identify all the risks related to this research that may arise in conducting this research, obtained the relevant ethical and/or safety approval (where applicable), and acknowledged my obligations and the rights of the participants.

Signature:  
Date: August 2018
Abstract

Sparse and low-rank models have been becoming fundamental machine learning tools and have wide applications in areas including computer vision, data mining, bioinformatics and so on. It is of vital importance, yet of great difficulty, to develop efficient optimization algorithms for solving these models, especially under practical design considerations of computational, communicational and privacy restrictions for ever-growing larger scale problems. This thesis proposes a set of new algorithms to improve the efficiency of the sparse and low-rank models optimization.

First, facing a large number of data samples during training of empirical risk minimization (ERM) with structured sparse regularization, the gradient computation part of the optimization can be computationally expensive and becomes the bottleneck. Therefore, I propose two gradient efficient optimization algorithms to reduce the total or per-iteration computational cost of the gradient evaluation step, which are new variants of the widely used generalized conditional gradient (GCG) method and incremental proximal gradient (PG) method, correspondingly. In detail, I propose a novel algorithm under GCG framework that requires optimal count of gradient evaluations as proximal gradient. I also propose a refined variant for a type of gauge regularized problem, where approximation techniques are allowed to further accelerate linear subproblem computation. Moreover, under the incremental proximal gradient framework, I propose to approximate the composite penalty by its proximal average under incremental gradient framework, so that a trade-off is made between precision and efficiency. Theoretical analysis and empirical studies show the efficiency of the proposed methods.
Furthermore, the large data dimension (e.g., the large frame size of high-resolution image and video data) can lead to high per-iteration computational complexity, thus results into poor-scalability of the optimization algorithm from practical perspective. In particular, in spectral k-support norm regularized robust low-rank matrix and tensor optimization, traditional proximal map based alternating direction method of multipliers (ADMM) requires to evaluate a super-linear complexity subproblem in each iteration. I propose a set of per-iteration computational efficient alternatives to reduce the cost to linear and nearly linear with respect to the input data dimension for matrix and tensor case, correspondingly. The proposed algorithms consider the dual objective of the original problem that can take advantage of the more computational efficient linear oracle of the spectral k-support norm to be evaluated. Further, by studying the sub-gradient of the loss of the dual objective, a line-search strategy is adopted in the algorithm to enable it to adapt to the Hölder smoothness. The overall convergence rate is also provided. Experiments on various computer vision and image processing applications demonstrate the superior prediction performance and computation efficiency of the proposed algorithm.

In addition, since machine learning datasets often contain sensitive individual information, privacy-preserving becomes more and more important during sparse optimization. I provide two differentially private optimization algorithms under two common large-scale machine learning computing contexts, i.e., distributed and streaming optimization, correspondingly. For the distributed setting, I develop a new algorithm with 1) guaranteed strict differential privacy requirement, 2) nearly optimal utility and 3) reduced uplink communication complexity, for a nearly unexplored context with features partitioned among different parties under privacy restriction. For the streaming setting, I propose to improve the utility of the private algorithm by trading the privacy of distant input instances, under the differential privacy restriction. I show that the proposed method can either solve the private approximation function by a projected gradient update for projection-friendly constraints, or by a conditional gradient step for linear oracle-friendly constraint, both of which improve the regret
bound to match the nonprivate optimal counterpart.
Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisor Professor Yiu-ming Cheung. I am very lucky to have him as my supervisor. His continuous guidance, insightful advice and research enthusiasm is the biggest support for my Ph.D. study in the past five years. I believe that I will benefit from what I have learned from him throughout my career in the future.

I would like to thank my research committee members, Dr. Byron Choi (my co-supervisor), Dr. Xiaowen Chu, and Dr. Li Chen for their generous advice and kind encouragement. I am also thankful to all the administrative (especially Kristina Tsang) and technical staffs in our department for their warm and generous help.

I would like to thank my academic siblings, Dr. Hong Jia, Dr. Qinmu Peng, Dr. Fangqing Gu, Mr. Sheungwai Chan, Mr. Yang Lu, Mr. Yiqun Zhang, Mrs. Yichao Zhou, Mr. Yang Zhou, Mr. Qiquan Shi and Mr. Meng Pang. Additionally, I want to thank all my friends in HKBU who come along with me through the years.

Finally, my parents have always been supportive. Their love makes me determined.
Table of Contents

Declaration i
Abstract ii
Acknowledgments v
Table of Contents vi
List of Tables xiv
List of Figures xv

Chapter 1 Introduction 1
  1.1 Background ......................................................... 1
  1.2 Motivations ......................................................... 4
    1.2.1 Large Scale Structured Sparse Empirical Risk Minimization . 4
    1.2.2 Robust Low Rank Matrix/Tensor Optimization ................. 4
    1.2.3 Differentially Private Optimization .......................... 5
  1.3 Main Contributions and Organization of this Thesis ............ 6

Chapter 2 Related Work 10
  2.1 Conditional Gradient Algorithms ................................. 10
    2.1.1 Basic Conditional Gradient Algorithm ........................ 10
    2.1.2 Generalized Conditional Gradient ............................ 11
    2.1.3 Approximate Linear Operator Evaluation ..................... 13
Chapter 2  Efficient Proximal Gradient Algorithms

2.2 Efficient Proximal Gradient Algorithms

2.2.1 Proximal Average

2.2.2 Incremental Gradient Descent Methods

2.3 Robust Low Rank Matrix Learning

2.3.1 Notation for Matrix

2.3.2 Robust Low Rank Subspace Learning

2.3.3 Spectral k-Support Norm

2.4 Robust Low Rank Tensor Learning

2.4.1 Notation for Tensor

2.4.2 Tensor Singular Value Decomposition (t-SVD) Algebraic Framework

2.4.3 Tensor Tubal Rank Definition and Tensor Nuclear Norm

2.5 Diﬀerentially Private Learning

2.5.1 Diﬀerentially Private Optimization for Feature-wise Distributed Dataset

2.5.2 Diﬀerentially Private Streaming Convex Optimization

Chapter 3  Efficient Generalized Conditional Gradient with Gradient Sliding for Composite Optimization

3.1 Introduction

3.2 Preliminary

3.2.1 Conditional Gradient Sliding Algorithm (CGS)

3.3 The Proposed Algorithm

3.3.1 General GCG-GS

3.3.2 Refined GCG-GS for Gauge Regularized Problem

3.4 Experiment

3.5 Summary
5.1 Introduction ................................................. 73
5.2 Preliminary: Scalable Algorithm with Spectral k-Support Norm ... 76
5.3 The Proposed Method ....................................... 77
   5.3.1 Formulation I: Usage with Squared Spectral k-Support Norm ... 78
   5.3.2 Formulation II: Usage with Spectral k-Support Norm .......... 82
   5.3.3 Algorithm .............................................. 84
5.4 Experiment .................................................. 89
   5.4.1 Synthetic Data ........................................... 90
   5.4.2 Background Modeling on Surveillance Videos .................... 93
   5.4.3 Face Reconstruction ...................................... 94
5.5 Summary ..................................................... 95

Chapter 6 Robust Low-Rank Tensor Minimization via a New Tensor Spectral k-Support Norm 96

6.1 Introduction .................................................. 96
6.2 Additional Notation ......................................... 102
6.3 A New Convex Relaxation for Robust Tensor Recovery: Tensor Spectral k-Support Norm .............................................. 102
   6.3.1 The New Tensor Spectral k-Support Norm ....................... 102
   6.3.2 Robust low-rank Tensor Recovery with TSP-k Norm ............ 105
6.4 An ADMM-based Optimization Algorithm for Medium-size Data ... 106
   6.4.1 Proximal Operator for TSP-k Norm-based Regularizer .......... 106
   6.4.2 Preconditioned ADMM-based Optimization Algorithm .......... 109
6.4.3 Computational Complexity .................................. 109
6.5 A Greedy Dual Optimization Algorithm for Larger-Size Data ...... 110
   6.5.1 Equivalent Dual Reformulation ................................ 111
   6.5.2 Polar Operator for TSP-k Norm ................................ 112
   6.5.3 Universal Primal-Dual Optimization Based Algorithm ......... 113
6.5.4 Complexity and Convergence Analysis .................................. 115
6.6 Experiment ............................................................................... 116
6.6.1 Experiments on medium size datasets ................................. 116
6.6.2 Experiments on Larger Size Datasets ................................. 122
6.7 Summary ................................................................................ 126

Chapter 7 Uplink Communication Efficient Differentially Private Sparse Optimization with Feature-wise Distributed Data 127

7.1 Introduction ............................................................................. 127
7.2 Additional Notation and Preliminary ...................................... 130
7.2.1 Sampling Distributions and Known Convergence Results for Block Coordinate Frank-Wolfe Algorithms .................................. 130
7.3 Block-Coordinate Frank-Wolfe under Arbitrary Sampling ........ 133
7.3.1 Algorithm Description ........................................................ 133
7.3.2 Expected Curvature ............................................................. 135
7.3.3 Convergence Analysis for BCFW with Arbitrary Sampling .................. 138
7.4 Uplink Communication Efficient Differentially Private BCFW with Distributed Features .................................................. 139
7.4.1 Algorithm Description ........................................................ 140
7.4.2 Analysis .............................................................................. 142
7.5 Summary ................................................................................ 144

Chapter 8 Differentially Private Streaming Convex Optimization with Decayed Privacy 146

8.1 Introduction ............................................................................. 146
8.2 Preliminary .............................................................................. 149
8.2.1 Problem Setup ..................................................................... 149
8.2.2 Differentially Private Follow The Approximate Leader ........ 149
8.3 Proposed Method ..................................................................... 151
### C.2 Proof Materials in Section 6.4

- C.2.1 Proof of Proposition 6.4.1
- C.2.2 Detailed derivation of preconditioned ADMM
- C.2.3 Computational Complexity Analysis for Algorithm 14

### C.3 Proof Materials in Section 6.5

- C.3.1 Proof of Proposition 6.5.1
- C.3.2 Proof of Proposition 6.5.2
- C.3.3 Proof of Proposition 6.5.3
- C.3.4 Proof of Corollary 6.5.1
- C.3.5 Line-search Subroutine
- C.3.6 Computational Complexity Analysis for Algorithm 16

### Chapter D Proofs of Materials in Chapter 7

- D.1 Preliminary
- D.2 Proofs of Materials in Section 7.3
  - D.2.1 Proof of Proposition 7.3.2
  - D.2.2 Proof of Proposition 7.3.3
  - D.2.3 Proof of Proposition 7.3.4
  - D.2.4 Proof of Proposition 7.3.5
  - D.2.5 Proof of Theorem 7.3.6
- D.3 Proofs of Materials in Section 7.4
  - D.3.1 Lemma D.3.1 and Proof: Private feature sharing
  - D.3.2 Lemma D.3.2 and Proof: Private index computing
  - D.3.3 Proof of Theorem 7.4.1
  - D.3.4 Proof of Theorem 7.4.2

### Chapter E Proofs of Materials in Chapter 8

- E.1 Additional Materials to Section 8.3.1
  - E.1.1 Window Tree Mechanism with Gamma Noise Perturbation
  - E.1.2 Window Tree Mechanism with Gaussian Noise Perturbation
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Summary of four real datasets</td>
<td>63</td>
</tr>
<tr>
<td>5.1</td>
<td>Videos used in the experiment</td>
<td>93</td>
</tr>
<tr>
<td>6.1</td>
<td>Recovery results on $n \times n \times n$ random data with different tubal rank</td>
<td>118</td>
</tr>
<tr>
<td>6.2</td>
<td>Comparison of PSNR values on example images from BSD</td>
<td>119</td>
</tr>
<tr>
<td>7.1</td>
<td>Comparison of BCFW convergence results</td>
<td>134</td>
</tr>
<tr>
<td>7.2</td>
<td>Summarization of Parameters in Theorem 7.4.2</td>
<td>142</td>
</tr>
<tr>
<td>8.1</td>
<td>Comparison of private COCO algorithms</td>
<td>150</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Illustration of the t-SVD in Eq. (2.4.31) ([73, 154]). Tensors from left to right are \( \mathbf{A}, \mathbf{U}, \mathbf{S} \) and \( \mathbf{V}^T \) ........................................................... 22

3.1 Objective function value versus total running time, polar operator time only, and gradient time only. ......................................................... 42

3.2 Objective function value versus number of outer iterations for different choice of inner loop count \( m \). ......................................................... 43

4.1 General convex loss with convex composite penalty: Empirical risk on training data versus effective passes of smooth hinge loss with graph-guided lasso on four real datasets. ..................................................... 64

4.2 General convex loss with convex composite penalty: Classification error on testing data versus effective passes and CPU time of smooth hinge loss with graph-guided lasso on two real datasets. ........................................................... 65

4.3 Strongly convex loss with convex composite penalty: Empirical risk on training data versus effective passes of logistic loss with large margin graph-guided lasso on four real datasets. ......................................................... 66

4.4 General convex loss with convex composite penalty: Test loss on testing data versus effective passes and CPU time of logistic loss with large margin graph-guided lasso on two real datasets. ......................................................... 67

4.5 Nonconvex composite penalty: Empirical risk on training data versus CPU time of least square loss with capped \( l_1 \) norm overlapping group lasso on synthetic datasets. ......................................................... 69
4.6 Empirical risk on training data versus effective passes with Nonconvex Graph-guided Lasso on four real datasets. ........................................ 70

4.7 Test loss on testing data versus effective passes with Nonconvex Graph-guided Lasso on four real datasets. ........................................ 71

5.1 Corruption ratio versus relative error. ........................................ 91

5.2 Rank ratio versus relative error. ........................................ 91

5.3 Data dimension versus CPU time. ........................................ 92

5.4 Background modeling results on campus and lobby dataset. ........ 93

5.5 Face reconstruction result on Extended yale B face dataset. ....... 95

6.1 Comparison of PSNR value for image denoising experiment on all images of Berkeley Segmentation Database. ................................. 117

6.2 Example images and recovery results. The indices of selected images from top to bottom are 006,020,087,144,193, where 020,087,144 and 193 are popularly demonstrated by previous related work. From left to right: original image, corrupted image; Recovered images by RPCA, PSSV, SNN, TNN, TSP-$k$. ........................................ 118

6.3 Comparison of PSNR value of each frame for simultaneous video completion and denoising experiment. ........................................ 119

6.4 Sample frames of simultaneous video completion and denoising experiment on videos: Bike, Basketball and Walking (from top to bottom). From left to right: original frame, missing and corrupted frame, recovered frame by TNN, recovered frame by TSP-$k$. ........................................ 120

6.5 Large size image recovery with 20% missing and 20% corruption. From top to bottom are images: bird, architecture, flower, human and tower. From left to right are: original image, input corrupted image, FWT output at 2000s CPU time, TNN and TSP-$k$ at 1000s CPU time, TNN and TSP-$k$ at 1500s CPU time, TNN and TSP-$k$ at 2000s CPU time. 122
6.6 Comparison of Large size image recovery with 20% missing and 20% corruption, which include $\ell_1$ (top) and $\ell_2$ (bottom) relative error versus CPU time. From left to right are results on images: Bird, Architecture, Flower, Human and Tower.

6.7 Sample frames of simultaneous high resolution video completion and denoising experiment. From left to right are sample frames of original Airport, Airport with corrupted and missing entries; Original Bridge, Bridge with corrupted and missing entries.

6.8 Comparison of relative error of the normalized video tensor against CPU time for simultaneous large scale video completion and denoising experiment. From left to right are $\ell_1$ and $\ell_2$ relative error decrease on the Airport, $\ell_1$ and $\ell_2$ relative error decrease on the Bridge.

6.9 Comparison of recovered frames. Rows from top to bottoms are: recovered frame of airport video by TNN, recovered frame of airport video by TSP-$k$ with greedy dual method; recovered frame of bridge video by TNN, recovered frame of bridge video by TSP-$k$ with greedy dual method. From left to right: frame recovered at CPU time 2000s, 3000s, 4000s, 5000s.
Chapter 1

Introduction

In this chapter, the research background is introduced in Section 1.1. Then, the motivations of this thesis are reported in Section 1.2. Section 1.3 summarizes main contributions and organization of this thesis.

1.1 Background

Sparse and low rank modeling have become a fundamental tool in machine learning, which also find wide applications in computer vision [154, 88, 87, 160], data mining [125, 83], bioinformatics [107] and so on. For computational concern and complexity control, we often consider their convex relaxation forms. Therefore, the sparse and low rank modelings can often be expressed as convex composite or constrained optimization problems. Facing large scale problems and complexity of the penalty that are prevalent nowadays, it is of vital significance to develop effective algorithms to solve such sparse and low rank optimization problems efficiently. In addition to computational efficiency, there are other design restrictions, including communication complexity in distributed optimization, privacy-protection in learning from sensitive individual data, ensuring non-degraded utility of the optimization results, all of which are making it ever challenging to the optimization algorithm design.

First order methods (also known as gradient descent methods) have been widely
used for sparse and low rank optimization due to their relatively low computation cost and scalability to large scale problems. Typical approaches include (generalized) conditional gradient methods and proximal/projected gradient methods.

(Generalized) Conditional Gradient Method, which have its origin date back to [10], have regained research interest nowadays due to low per-iteration cost. The low per-iteration complexity mainly comes from the relatively simple linear subproblem required to solve on each iteration. This linear subproblem is derived from the duality gap ([144]; [14]), which is related to necessary optimal condition and is always an upper bound of the primal gap. Also, the computation cost of the linear subproblem can be further reduced by approximation techniques ([80]; [144]). One drawback of the conditional gradient based methods is that the associated convergence rate is often an order inferior to the optimal rate for first order methods, i.e. accelerated proximal gradient methods in the next subsection.

Proximal/Projected Gradient Methods solves a quadratic subproblem called proximal map in each iteration. Accelerated full gradient methods ([103]; [7]) can achieve optimal convergence rate of the first order methods. Facing large scale problems, we often prefer its stochastic variant which only evaluates gradient based on single data sample on each iteration. Examples include: SGD ([11]), which is the most simple stochastic methods that samples one data at each iteration; RDA ([139]) averages all past gradients; accelerated stochastic gradient methods like ([43]) is the stochastic variant of optimal gradient method. However, stochastic gradient methods often have sublinear convergence rate which is inferior to what full gradient method is possible to achieve. Recently, new incremental gradient methods have been proposed that possess both scalability and fast convergence property. These methods only calculate gradients associated with a randomly picked data sample in each iteration as stochastic gradient methods, thus have comparable low per-iteration computation cost. More importantly, by well exploiting the finite sum structure of the loss function which stochastic methods do not, these incremental methods are able to achieve linear convergence rate as full gradient methods. For example,
SAG ([116]) uses the average of the stored past gradients, one for each data. SVRG ([66], [140]) adopts a multi-stage scheme to progressively controls the variance of the stochastic gradient. Both methods have linear convergence rate for strongly convex problem. But with the theoretical convergence result for general convex loss is unclear. SAGA ([25], [26]) has both sublinear convergence guarantee for general convex loss and linear convergence for strongly convex loss. It is a midpoint of SAG and SVRG by taking both update pattern from them in its iteration. There are also other incremental methods like FINITO ([28]) and MISO ([91]), which consumes more memory in that they not only store the gradient but also the variable. S2GD ([76]) is a method very similar to SVRG with difference only in stage length. SDCA ([120]) is a dual incremental method.

Due to their relatively low per-iteration computational cost, fast convergence rate and effectively handling various regularization/constraints in sparse and low rank optimization, I will focus on these two types of algorithms in this thesis and develop better algorithms based on them to deal with the emerging computation requirements of larger amount of training samples, growing size of data dimensions and differentially private restrictions.

**Differential Privacy:** Modern machine learning models often optimizes the model from sensitive data that are collected from individuals. To avoid breaching the privacy of the individuals, privacy protection mechanism have been considered to ensure that the adversary cannot infer any individual data from the output of the learning process. Beginning with the seminal work [21], which considers private ERM training under the formal statistical differential privacy notion [35], various differentially private optimization algorithms have been developed for training the model with centralized datasets [124] [72] [5] [128] [32] [68] and sample-wise distributed datasets [57] [104] [47].
1.2 Motivations

1.2.1 Large Scale Structured Sparse Empirical Risk Minimization

When the amount of data samples for training becomes large, the gradient evaluation becomes expensive in computation. In particular, for structured sparsity regularized ERM optimization, GCG and PG are two popular lines of methods. For the GCG method, recall from the previous section that the inferior iteration complexity leads to extra amount of gradient evaluations. Although the per-iteration cost for LO can be small enough to afford excess iterations, the increased demand for gradient evaluation can raise an inevitable trade-off, especially for large scale problems. To improve gradient evaluation efficiency, I propose a novel algorithm that requires an optimal count of graduate evaluations as proximal gradient. As for the PG method, the fast incremental proximal gradient methods mentioned are only applicable to problems regularized by simple penalty. In order to reduce per-iteration gradient evaluation cost, I therefore propose to approximate the composite penalty by its proximal average ([6];[146];[157]) under incremental gradient framework.

1.2.2 Robust Low Rank Matrix/Tensor Optimization

When the dimension of the input data becomes large, the high per-iteration complexity become the bottleneck of the optimization algorithm. In particular, in robust low rank matrix/tensor optimization, the spectral k-support norm [36;95;96] has been proposed to prompt low rankness of matrix by applying the k-support norm to the singular values of the matrix. Compared with nuclear norm, it provides tight relaxation of the rank k matrices under unit $\ell_2$ norm ball of its singular values rather than infinite norm ball, which is often more preferred [36;95]. Despite the superior recovery performance compared with other convex relaxations like nuclear norm, the spectral k-support norm is much more difficult to be optimized, which therefore severely limits its application domain, particularly for big data analysis.
Although methods developed for k-support norm that relies on proximal map of the squared k-support norm \[ \text{[3 36 80]} \] can be migrated to the spectral k-support norm, its computation is laborious. A major reason is the full SVD decomposition involved in the proximal mapping computation. Furthermore, restricted by the property of the k-support norm, efficient approximation methods for nuclear norm (e.g. power method and Lanczos method) that requires leading singular values only are hardly applicable to spectral k-support norm. Further, a search operation that segments singular values into certain groups also needs additional computation.

### 1.2.3 Differentially Private Optimization

In sparse optimization, we are often dealing with large amount of sensitive data collected from individuals, which raises the privacy protection restriction. In addition to the traditional centralized optimization, distributed and streaming optimizations are two pervasive scalable computing scenarios widely encountered in sparse learning.

**Distributed private sparse optimization:** Beginning with the seminal work \[ \text{[21]} \], which considers private ERM training under the formal statistical differential privacy notion \[ \text{[35]} \], various differentially private optimization algorithms have been developed for training the model with centralized datasets \[ \text{[121 72 5 28 62 68]} \] and sample-wise distributed datasets \[ \text{[37 101 17]} \]. However, the feature-wise distributed dataset setting is under exploitation by existing literature. Such setting appears in many real applications, where the information describing an individual is collected and held by different parties which can be different sets of sensory systems or different organizations. For example, a person’s medical records are sensitive personal information that can be held by several clinics. Although privacy issue has been considered for these vertically-partitioned datasets \[ \text{[48 143 93]} \], it has rarely been studied with the more restrict differential privacy notion. It would be ideal to make use of all attributes kept by different parties in a distributed fashion, while still ensuring differential privacy.

**Steaming private sparse optimization:** Within convex learning, despite
the various offline DP algorithms [21] that adapt to different problem structures, e.g. private gradient descent for unconstrained function [21], private projected gradient (PG) descent for projection-friendly constraint sets [5], or private conditional gradient (CG) for linear oracle-friendly constraint sets [29], private COCO algorithms [61, 130, 63] are relatively less studied. With increasing number of instances, the existing private convex optimization with constraints accumulate regret faster than nonprivate optimal counterparts, which inspires us the growing price of utility to pay for privacy protection for every streaming-in individual instance over time. Also, they rely exclusively on the projected gradient (PG) step for updating the variable provided that the constraints are projection-friendly, regardless of the existence of many widely applied models yet with linear oracle-friendly constraint sets, i.e. suiting a CG step better than a PG step, see [60]; [42].

1.3 Main Contributions and Organization of this Thesis

The remaining chapters of the thesis are organized as follows:

In Chapter 2, I provide a literature review of related work on sparse and low rank optimization, including (generalized) conditional gradient method, incremental gradient method, proximal/projected gradient method. I also review the robust low rank matrix and tensor modeling techniques, including the low rank inducing regularizers, tensor singular value decomposition (t-SVD) framework. The formal definition of differential privacy and its associated techniques are also reviewed.

In Chapter 3, I consider the unconstrained composite optimization task, which is a suitable model for many sparse optimization tasks. I propose a novel algorithm called Generalized Conditional Gradient with Gradient Sliding (GCG-GS). GCG-GS requires an optimal count of graduate evaluations as proximal gradient. I also propose a refined variant for a type of gauge regularized problem where approximation techniques are allow to further accelerate linear subproblem computation. Experiments of
a group sparse regularized CUR-like matrix factorization problem on four real-world datasets demonstrate the efficiency of our method.

In Chapter 4, I focus on empirical risk minimization (ERM) regularized by a complex composite penalty. In detail, to make proximal average (PA) an ideal technique for optimizing ERM with composite penalties, this chapter proposes a new PA-based algorithm called IncrePA by incorporating proximal average approximation into an incremental gradient framework. The proposed method is a more optimal PA-based method that features lower per-iteration cost, a faster convergence rate for convex composite penalties, and guaranteed convergence for even nonconvex composite penalties. Experiments on both synthetic and real datasets demonstrate the efficacy of the proposed method in optimizing convex and nonconvex ERM with composite penalties.

In Chapter 5, I focus on robust low rank subspace learning, which is to recover a low rank matrix under gross corruptions that are often modeled by another sparse matrix. Within this learning, I investigate the spectral k-support norm, a more appealing convex relaxation than the popular nuclear norm, as a low rank penalty in this chapter. Despite the better recovering performance, the spectral k-support norm entails the model difficult to be optimized efficiently, which severely limits its scalability from the practical perspective. Therefore, this chapter proposes a scalable and efficient algorithm which considers the dual objective of the original problem that can take advantage of the more computational efficient linear oracle of the spectral k-support norm to be evaluated. Further, by studying the sub-gradient of the loss of the dual objective, a line-search strategy is adopted in the algorithm to enable it to adapt to the Hölder smoothness. Experiments on various tasks demonstrate the superior prediction performance and computation efficiency of the proposed algorithm.

In Chapter 6, I propose a new tensor norm by first re-deriving the TNN based on yet another tubal rank quantity, which unifies the existing two different TNN definitions. For the same rank quantity but by an alternative convex relaxation, I
then propose the new tensor spectral k-support norm (TSP-$k$), which interpolates between TNN and tensor Frobenius norm. Together, the TNN phase drives minor single values to zero to induce a low-rank tensor, while the tensor Frobenius norm component captures more global information of the tensor for better preserving the intrinsic structure. I consider the robust tensor minimization problems to exemplify the usage of TSP-$k$ for low-rank tensor regularization. I provide two optimization algorithms for both primal and dual forms of the objective function, of which the primal method is effective for medium tensors while the dual one is in essence greedy thus scales to larger size tensors. I derive the proximal operator and polar operation for TSP-$k$ as two key computation components of the corresponding optimization procedure. Experiments on synthetic, image and video datasets in medium and large sizes, all verify the superiority of TSP-$k$ norm and the effectiveness of the both optimization methods in comparison with the existing counterparts.

In Chapter 7, I study a nearly unexplored context with features partitioned among different parties under privacy restriction. Motivated by the nearly optimal utility guarantee achieved by centralized private Frank-Wolfe algorithm [129], I develop a distributed variant with guaranteed privacy, utility and uplink communication complexity. To obtain these guarantees, we provide a much generalized convergence analysis for the block-coordinate Frank-Wolfe method under arbitrary sampling, which greatly extends known convergence results that are only applicable to two specific block sampling distributions. I also design an active feature sharing scheme by utilizing private Johnson-Lindenstrauss transform, which is key to updating local partial gradients in a differentially private and communication efficient manner.

In Chapter 8, I focus on differentially private restricted streaming machine learning and data mining problems, where individual data are collected and revealed consecutively. These problems can often be modeled and solved under the Constrained Online Convex Optimization (COCO) algorithmic framework. The ever-growing amount of sensitive individual data is posing greater challenge to the contradictory goals of privacy protection and reasonable model usability. In this chapter, I formally
investigate whether we can improve the utility of the private COCO by trading the privacy of distant input instances, under the statistical notion of differential privacy (DP). My method protects decayed privacy by adapting a window tree mechanism for maintaining a private gradient summation, which is then used to construct an approximation function for updating the new response variable at each timestamp. I then show that we can either solve the private approximation function by a projected gradient (PG) update for projection-friendly constraints, or by a conditional gradient (CG) step for linear oracle-friendly constraint, both of which improve the regret bound to $O(\ln T)$ with respect to sequence length $T$, matching the nonprivate optimal regret. In particular, the CG-based variant is the first known private COCO designed for problems with LO-friendly constraint sets, and thus broadens the applicability of COCO with privacy guarantee.

Finally, Chapter 9 draws a conclusion for this thesis and discusses some potential directions of future work. Proofs of the theoretical results and the related publication list can be found in appendix.
Chapter 2

Related Work

2.1 Conditional Gradient Algorithms

2.1.1 Basic Conditional Gradient Algorithm

The classic CG method is designed to solve constrained convex optimization problems. This method is generally composed of (sub)gradient evaluation, linear oracle (LO) evaluation, variable update and additional refinement based on the geometry of the constrained set or some other local improvement. The constrained convex optimization is as follows,

$$\min_x l(x), \ s.t \ x \in \mathcal{C},$$

(2.1.1)

where $l(x)$ is often called the loss function and $\mathcal{C}$ is the constrained set.

The most basic CG algorithm under convenient assumptions like $l(x)$ is $L$-smooth, which is presented in Algorithm [I]. We use standard notation unless otherwise stated.
Algorithm 1: Basic FW method for smooth $l(x)$ constrained on closed convex set $C$

Input: $x_0, \gamma_t$

1: \begin{algorithmic}
   \STATE \textbf{for} $t = 0, 1, ..., T - 1$ \textbf{do}
   \STATE Evaluate Gradient $\nabla l(x_t)$;
   \STATE Evaluate Linear Oracle: $s_t = \arg\min_{s \in C} \langle s, \nabla l(x_t) \rangle$;
   \STATE Update by step size $\gamma_t = \frac{2}{t+1}$: $x_{t+1} = (1 - \gamma_t)x_t + \gamma ts_t$;
   \STATE \textbf{end for}

Output: $x_K$

This method converges with $O\left(\frac{1}{T}\right)$ for general convex $l(x)$. It converges slower than projection or proximal mapping based first order algorithms under same condition. However, due to the lower cost LO step, for some constraint set $C$ we can expect a tradeoff between per-iteration cost and iteration complexity. Also, the FW method shows better scalability for large scale problems, thanks to the lower per-iteration cost. According to Algorithm 1, we can summarize the main steps of the FW algorithm into three main parts: gradient evaluation (line 2), linear oracle evaluation (line 3), updating $x$ with proper step size $\gamma$ (line 4). Also, some additional local refinement can be adopted to improve its efficiency.

By exploiting the structure of some particular problems, some authors also propose to integrate the FW method with other type of methods, e.g. mirror proximal method, Universal Primal-Dual method, Proximal Gradient Method. The resulting algorithms make full advantage of the scalability of the FW type method (especially the low cost LO oracle), as well as the efficiency gain by exploiting the problem structure.

2.1.2 Generalized Conditional Gradient

Generalized conditional gradient method (GCG) is suitable for the unconstrained composite optimization problems of the form

$$\min_{x \in \mathcal{X}} F(x) = l(x) + r(x), \quad (2.1.2)$$
where \( l(x) \) is a smooth convex loss function, and \( r(x) \) stands for a nonsmooth convex regularizer, \( \mathcal{X} \) is a general vector space equipped with inner product (i.e. \( \mathbb{R}^d \)).

GCG is a generalization to classic conditional gradient method (CG) \([40]\), which solves constraint problem on a subset of \( \mathcal{X} \), i.e. convex compact set \([60]\). Also, to keep the nonsmooth part intact, it defines the following alternative duality gap at iteration \( k \) (\([144]; [13]\)):

\[
G(x_k) = l(x_k) + r(x_k) - \inf_{x \in \mathcal{X}} \{(l(x_k) + \langle x - x_k, \nabla l(x_k) \rangle + r(x)) \};
\]

\[
= r(x_k) - \inf_{x \in \mathcal{X}} \{ r(x) + \langle x - x_k, \nabla l(x_k) \rangle \}
\]

\[
= \sup_{x \in \mathcal{X}} \{ r(x_k) - r(x) - \langle x - x_k, \nabla l(x_k) \rangle \},
\]

instead of taking the sub-di\(^\text{erential} \) of nonsmooth \( r(x) \) as some CG methods proposed \([60]\). The duality gap is essential for constructing the linear operator, because it is related to the optimal necessary condition and is always an upper bound approximation to the primal gap \( F(x) - F(x^*) \), as detailed in the following lemma.

**Lemma 2.1.1.** (\([144], \text{Proposition 4}\))

For problem \( 2.1.2 \) with any \( x \in \mathcal{X} \), the duality gap \( G(x) \geq 0 \) and \( G(x) = 0 \) iff \( x \) satisfies the necessary optimal condition\(^1\). Also, the duality gap is always an upper bound of the primal gap that \( G(x) \geq F(x) - F(x^*) \), where \( x^* \) denotes the global optimal point.

Hence, GCG solves problem \( 2.1.2 \) by minimizing the duality gap on each iteration, which amounts to evaluating a linear operator (LO),

\[
d_k = \arg \max_{x \in \mathcal{X}} r(x_k) - r(x) - \langle x - x_k, \nabla l(x_k) \rangle;
\]

\[
\Leftrightarrow d_k = \arg \min_{x \in \mathcal{X}} \langle x, \nabla l(x_k) \rangle + r(x)
\]

Note that additional assumptions should be made to \( F(x) \), otherwise the above linear subproblem may diverge. To avoid introducing additional complexity, in this thesis we adopt the assumption that the solution sequences \( d_k \) and \( x_k \) (also \( a_t \) and

\(^1\)We say \( x \) satisfies the necessary optimal condition if \( 0 \in \nabla l(x) + \partial r(x) \).
to be introduced later) are finite whose maximum distance between each other is upper bounded by a positive constant $D_s$. This assumption is same as Assumption 3 in [144], where more sophisticated equivalent assumptions were also discussed (see Proposition 3 in [144]).

Then the next step $x_{k+1}$ can be obtained by

$$x_{k+1} = (1 - \alpha_k)x_k + \alpha_k d_k,$$

(2.1.5)

where the step size $\alpha_k$ can be set to deterministic sequence for example of order $O(\frac{1}{k})$ or choosing by optimizing the following problem,

$$\alpha_k = \arg \min_{\alpha \in [0,1]} F((1 - \alpha)x_k + \alpha d_k).$$

(2.1.6)

GCG method has $O(\frac{1}{K})$ convergence rate for problem (3.1.1), where $K$ is the total number of iterations. Or equivalently speaking, it needs $O(\frac{1}{\epsilon})$ iterations to find an $\epsilon$ accurate solution.\footnote{The $\epsilon$ accurate solution refers to a solution $x$ having the primal gap no larger than $\epsilon$.} Apparently, this incurs additional count of gradient evaluation than their PG counterpart based on this one gradient evaluation per-iteration scheme.

### 2.1.3 Approximate Linear Operator Evaluation

The linear operator (2.1.4) admits low per-iteration cost that allows them to afford excess count of iterations, which is the major motivation for adopting GCG method rather than PG method for some tasks. This evaluation admits various approximation techniques. Specifically we consider the case when $r(x)$ is the so called generalized gauge function [144] defined as

$$r(x) = h(\kappa(x)),$$

(2.1.7)

where $h(\cdot)$ is a convex increasing function and $\kappa(x)$ is a gauge function (convex, positively homogeneous) satisfying

$$\kappa(a) = \inf \{ \rho : a \in \rho \mathcal{C} \}.$$  

(2.1.8)
The convex compact subset $C$ is the unit ball of the gauge $\kappa(x)$, namely

$$C = \{a \in \mathcal{X} : \kappa(a) \leq 1\}. \quad (2.1.9)$$

This formal definition of $r(x)$ is a little complicated, indeed one can simply take it as a generalized norm function.

The key idea to accelerate LO evaluation is to circumvent direct computation of $\kappa(x)$ by approximation techniques. We first prepare ourselves with some conversion following [144]. Convert (2.1.4) to a constrained form, it gives

$${d_k} = \arg \min_{x : r(x) \leq \zeta} \langle x, \nabla l(x_k) \rangle. \quad (2.1.10)$$

To avoid estimating $\zeta$, it derives $d_k$ by direction and scalar separately. For estimating the direction $a_k$, it follows

$${a_k} = \arg \min_{a : \kappa(a) \leq 1} \langle a, \nabla l(x_k) \rangle = \arg \min_{a \in \mathcal{C}} \langle a, \nabla l(x_k) \rangle. \quad (2.1.11)$$

We leave the derivation of scalar later when used.

The approximation for accelerating the evaluation of LO lies in two parts. The first is to allow $d_k$ or $a_k$ to be approximately calculated, namely $a_k$ is relaxed to satisfy,

$$\langle a_k, \nabla l(x_k) \rangle \leq \epsilon_k + \min_{a \in \mathcal{C}} \langle a, \nabla l(x_k) \rangle. \quad (2.1.12)$$

This approximation can further reduce the per-iteration cost of LO evaluation and reduction of overall runtime is observed practically [60]. However, with such approximation, the iteration complexity guarantee is weakened by a factor associated with the degree of approximation allowed [59]. As a result, the count of gradient evaluations is increased which leads to an inevitable trade-off between the time increase of gradient evaluation and the decrease of LO evaluation, especially for large scale problems whose gradient is expensive to compute.

The other approximation is related to the constraint set. When $C$ is a convex hull of atomic domain $A$, (2.1.11) can be equivalently solved on mathcal$A$,

$$\min_{a \in \mathcal{C}} \langle a, \nabla l(x_k) \rangle = \min_{a \in \mathcal{A}} \langle a, \nabla l(x_k) \rangle = -\kappa^o(-\nabla l(x_k)), \quad (2.1.13)$$

14
where in the last equation we have used the notation \( \kappa^o(-\nabla l(x_k)) = \max_{a \in A} \langle a, -\nabla l(x_k) \rangle \) called polar operator \([152]\). Hence, the evaluation of LO is converted to the evaluation of the polar operator, which is more efficient to deal with than \( \kappa(x) \) itself and is the actual form adopted practically. A simple example of this polar operator is when \( \kappa(x) \) being a norm, then we can immediately find the associated \( \kappa^o(g) \) is its dual norm.

2.2 Efficient Proximal Gradient Algorithms

This section gives an overview of PA technique and incremental gradient framework.

2.2.1 Proximal Average

PA \([6, 146]\) has been recently introduced to deal with composite regularizers. It admits a compact calculation when each single component satisfies Assumption 4.2.4. PA only requires each component of \( r(x) \) has simple proximal map, even when it is computationally expensive for \( r(x) \) itself. The following definition describes the PA \( \hat{r}(x) \) of \( r(x) \).

**Definition 2.2.1.** (PA \([6, 146]\)) The PA of \( r \) is the unique semicontinuous convex function \( \hat{r}(x) \) such that \( M^n_{\hat{r}(x)} = \sum_{k=1}^{K} \alpha_k M^n_{r_k} \). The corresponding proximal map of the PA \( \hat{r}(x) \) is

\[
P^n_{\hat{r}}(x) = \sum_{k=1}^{K} \alpha_k P^n_{r_k}(x).
\] (2.2.14)

Therefore, once approximating \( r(x) \) by \( \hat{r}(x) \), we can obtain the proximal map of \( \hat{r}(x) \) by simply averaging the proximal map of each constituent regularizer \( r_k(x) \). The next lemma shows that the approximation of \( \hat{r}(x) \) can be controlled arbitrarily close to \( r(x) \) by the step size \( \eta \).

**Lemma 2.2.2.** (\([146]\)) Under Assumption 4.2.4, we have \( 0 \leq r(x) - \hat{r}(x) \leq \frac{\eta \bar{M}^2}{2} \), where \( \bar{M}^2 = \sum_{k=1}^{K} \alpha_k M_k^2 \).
In fact, although [146] verifies the above lemma provided that \( r_k(x) \) is convex, GD-PAN shows that it actually applies to nonconvex cases as long as Assumption 2 holds.

### 2.2.2 Incremental Gradient Descent Methods

The incremental gradient methods proposed recently make an improvement on stochastic gradient methods provided that the training data is finite. Generally, at each iteration, these methods approximate the full gradient by a combination of a random gradient evaluated at the latest variable with past gradients. There are several types of incremental gradient methods. For example, SAG utilizes a gradient table to record past gradients for each data sample index. SVRG uses a single full gradient evaluated periodically. Both of the methods have linear convergence for strongly convex and smooth problem. SAGA shares part of update pattern from both SAG and SVRG, and has theoretical guarantee for both general convex and strongly convex problem.

Denote the variable table at iteration \( t \) by \( \phi_t \), which contains \( N \) vectors recording the iterate \( x_t \) in a randomly-select-and-replace strategy. That is, the algorithm randomly selects an index \( i_t \) from 1 to \( N \) and then replaces the \( i^{t+1} \)-th column of \( \phi_t \) by the latest iterate \( x_t \), i.e.

\[
\phi_i^{t+1} = \begin{cases} 
\nabla x_t, & i = i_t \text{ (Replace)} \\
\nabla \phi_i^t, & i \neq i_t \text{ (Unchanged)}. 
\end{cases}
\]  

(2.2.15)

Let \( \nabla l_i(\phi_i^t)(i = 1, 2, ..., N) \) be the gradient table. SAGA, like SAG, updates the random \( i^{t} \)-th gradient with \( \nabla l_i(x^t) \) while keeping other terms unchanged:

\[
\nabla l_i(\phi_i^{t+1}) = \begin{cases} 
\nabla l_i(x^t), & i = i^{t} \text{ (Replace)} \\
\nabla l_i(\phi_i^t), & i \neq i \text{ (Unchanged)}. 
\end{cases}
\]  

(2.2.16)

Hence, we only need to evaluate the gradient related to the \( i^{t} \) data sample by computing \( \nabla l_i(x^t) \). Also, the variable table \( \phi_t \) is introduced for notational convenience and thus need not be explicitly stored.
Based on the stored gradient table, SAG proposes to construct a variance reduced gradient estimation by averaging the gradient table, i.e. \( G_t = \frac{1}{N} \sum_{i=1}^{N} \nabla l_i(\phi_i^t) \). On the contrary, SVRG proposes to use the unbiased estimation \( G_t = \nabla l_i^t(x^t) - \nabla l_i^t(\tilde{x}_s) + \frac{1}{n} \sum_{i=1}^{N} \nabla l_i(\tilde{x}_s) \), where \( \frac{1}{n} \sum_{i=1}^{N} \nabla l_i(\tilde{x}_s) \) is the batch gradient evaluated periodically on \( \tilde{x}_s \) (e.g. every \( 2N \) iterations). SAGA propose to approximate the gradient for iteration \( t \):

\[
G_t = \nabla l_i^t(\phi_i^{t+1}) - \nabla l_i^t(\phi_i^t) + \frac{1}{n} \sum_{i=1}^{N} \nabla l_i(\phi_i^t).
\] (2.2.17)

SAGA shows that this gradient estimation strategy actually stands in middle of that used by SAG and SVRG. Also, conditioned on information up to the \( t \)-th iteration, \( G_t \) is an unbiased estimation of the full gradient in expectation. According to ([66], [140]), such approximate gradients have the reduced variance, which would lead to speed up over stochastic methods. SAGA admits iteration schemes involving proximal mapping, but only for simple penalty functions equipping closed-form update and is incapable to handle the more complex composite penalties.

### 2.3 Robust Low Rank Matrix Learning

#### 2.3.1 Notation for Matrix

For a matrix \( X \), \( \|X\|_1, \|X\|_{2,1}, \|X\|_F, \|X\|_* \) denote its \( \ell_1, \ell_{2,1} \) (sum of \( \ell_2 \) norm of each column), Frobenius and nuclear norm (sum of singular values) correspondingly. For a particular singular value decomposition (SVD) of matrix \( X \in \mathbb{R}^{m,n} \), we denote it as \( X = U \text{diag}(\sigma)V^T \), where \( \sigma = (\sigma_1, ..., \sigma_{\min(m,n)}) \) is the vector formed by singular values arranged in nonincreasing order and \( \text{diag}(\sigma) \) is the diagonal matrix with its \( i \)-th diagonal element being \( \sigma_i \). For a function \( f \), we use \( \nabla f(\Gamma) \) to denote its gradient or one of its subgradient at \( \Gamma \), and use \( \partial f(\Gamma) \) to denote the set of subgradient at \( \Gamma \). The superscript \( (\cdot)^T \) denotes the transpose for a matrix or the adjoint operation for a linear map.
2.3.2 Robust Low Rank Subspace Learning

In general, robust subspace learning methods seek a low rank component $L$ plus a sparse component $S$ capturing grossly corrupted outliers. $L$ and $S$, together with a constant matrix $M$, are related by a linear constraint with constant linear map $B$, which can be summarized into the following nonsmooth linear constraint problem,

$$\arg \min_{L,S} \|L\|_r + \lambda \|S\|_s, \quad s.t. \ B(M - L) = S,$$

(2.3.18)

where the penalty $\| \cdot \|_r$ is used to promote low rankness of $L$, which is chosen as the spectral k-support norm $\| \cdot \|_{sp,k}$ in Chapter 5 of this thesis. The second term $\| \cdot \|_s$ is the sparsity inducing penalty which can be $\ell_1$ or $\ell_{2,1}$ norm. $\lambda$ is a constant parameter used to balance low rankness and sparsity.

In Chapter 5 of this thesis, we focus on the RPCA as a practical application, where $M$ is the input data matrix $D$ and $B$ is identity matrix. With spectral k-support norm and $\ell_1$ norm, the RPCA problem can be formulated as

$$\arg \min_{L,S} \|L\|_{sp,k} + \|S\|_1, \quad s.t. \ D - L = S \ (\text{RPCA}).$$

(2.3.19)

2.3.3 Spectral k-Support Norm

We first recall the k-support norm, which is introduced by as a convex surrogate of the nonconvex cardinality function (a.k.a. $\ell_0$ norm) for sparsity vector prediction. observes that the most popular $\ell_1$ norm is the convex hull of $\ell_0$ norm on unit $\ell_\infty$ ball which assumes each entry to be bounded,

$$\text{conv}(x \in \mathbb{R}^d | \|x\|_0 \leq k, \ |x|_\infty \leq 1).$$

(2.3.20)

However, in many cases, we prefer the $\ell_2$ norm of $x$ to be bounded, i.e.

$$\text{conv}(x \in \mathbb{R}^d | \|x\|_0 \leq k, \ |x|_2 \leq 1),$$

(2.3.21)

which can help improve robustness and generalization. In this perspective, proposes the k-support norm which can be calculated as follows,

$$\|x\|_{sp,k} = \left( \sum_{i=1}^{k-t-1} (x_i)^2 + \frac{1}{t+1} \left( \sum_{i=k-t}^{d} x_i \right)^2 \right)^{\frac{1}{2}},$$

(2.3.22)
where \( t \) is an index satisfying the following relationship,

\[
x_{k-t-1} > \frac{1}{t+1} \sum_{i=k-t}^{d} x_i \geq x_{k-t}.
\]  
(2.3.23)

95, 96 then extend the k-support norm to low rank promoting purpose for matrices. Similar to the definition of nuclear norm, the spectral k-support norm (we use the same notation \( \| \cdot \|_{sp,k} \) as the spectral form when the variable is matrix) is also defined in terms of the matrix singular values and is thus unitary invariant. In detail, for a matrix \( Z \in \mathbb{R}^{m \times n} \) and denoting a particular singular value decomposition (SVD) as \( Z = U \text{diag}(\sigma) V^T \), the spectral k-support norm can be computed by

\[
\|Z\|_{sp,k} = \left( \sum_{i=1}^{k-t-1} (\sigma_i)^2 + \frac{1}{t+1} \left( \sum_{i=k-t}^{\min\{m,n\}} \sigma_i \right)^2 \right)^{\frac{1}{2}}
\]  
(2.3.24)

where index \( t \in \{0, 1, ..., k-1\} \) is searched to satisfy \( \sigma_{k-t-1} > \frac{1}{t+1} \sum_{i=k-t}^{\min\{m,n\}} \sigma_i \geq \sigma_{k-t} \).

Apparently, the unit ball of spectral k-support norm is defined in terms of its singular values and can be expressed as the convex hull of vectors with at most \( k \) cardinality lying within the \( \ell_2 \) norm ball, i.e.

\[
\mathcal{D} = \text{conv}(\mathcal{A}), \text{ where } \mathcal{A} = \{ A \in \mathbb{R}^{(m,n)} | A = U \text{diag}(\sigma) V^T, \|\sigma\|_0 \leq k, \|\sigma\|_2 \leq 1 \}. 
\]  
(2.3.25)

When \( k = 1 \), the spectral k-support norm becomes nuclear norm, and when \( k = \min\{m, n\} \), it coincides with Frobenius norm. Intuitively, it penalizes the largest \( k-t-1 \) singular values with \( \ell_2 \) norm while penalizing smaller \( t+1 \) singular values with \( \ell_1 \) norm. The k-support norm and spectral k-support norm are indeed norm functions 93, 95. Also, denoting the dual norm by \( \| \cdot \|_{sp,k}^* \), for any matrix \( Z \) with a particular SVD of \( Z = U \text{diag}(\sigma) V^T \), we have

\[
\|Z\|_{sp,k}^* = \sqrt{\sum_{i=1}^{k} \sigma_i^2}.
\]  
(2.3.27)

It is obvious that the dual norm can be more efficient to compute because: 1) it only requires the first \( k \) singular values; 2) it avoids search for index \( t \).
also generalizes the spectral k-support norm to the so-called spectral (k,p)-support norm by using $\ell_p$ unit norm ball constraint in eq. (2.3.24) instead of the $\ell_2$ unit norm ball. This extension can be denoted by $|| \cdot ||_{sp,(k,p)}$, under which the spectral k-support norm is $|| \cdot ||_{sp,(k,2)}$. shows that by varying $p$, the generalized spectral (k,p)-norm can capture the decay of singular values of the desired low rank matrix in a low rank matrix completion task. Most computation of the spectral (k,p)-support norm is similar with spectral k-support norm. For example, to calculate the dual norm, we simply change 2 with $q$ by

$$||Z||^*_{sp,(k,p)} = \left( \sum_{i=1}^{k} \sigma_i^p \right)^{\frac{1}{q}}, \text{ where } \frac{1}{p} + \frac{1}{q} = 1. \tag{2.3.28}$$

### 2.4 Robust Low Rank Tensor Learning

#### 2.4.1 Notation for Tensor

We denote tensors by boldface Euler letters, e.g. $\mathcal{A}$. For a third order tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we use the MATLAB notation $\mathcal{A}(; ; , i), \mathcal{A}(; , i , :) , \mathcal{A}(i , ; , :)$, to denote the $i$-th frontal, lateral and horizontal correspondingly. For brevity, we also denote the $i$-th frontal slice by $\mathcal{A}^{(i)}$ and the $(i, j, k)$-th entry by $\mathcal{A}_{ijk}$. Let $D = \min\{n_1, n_2\} \cdot n_3$.

#### 2.4.2 Tensor Singular Value Decomposition (t-SVD) Algebraic Framework

We begin the introduction of the t-SVD algebraic framework with the following tensor-tensor product definition:

**Definition 2.4.1. (Tensor Product (t-Product))** The t-Product between tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$ is defined as $\mathcal{A} \ast \mathcal{B} = \mathcal{C} \in \mathbb{R}^{n_1 \times n_4 \times n_3}$ with the $(i, j)$-th tube $\hat{c}_{ij}$ of $\mathcal{C}$ computed as

$$\hat{c}_{ij} = \mathcal{C}(i, j, :) = \sum_{k=1}^{n_2} \mathcal{A}(i, k, :) \ast \mathcal{B}(k, j, :), \tag{2.4.29}$$
where * denotes the circular convolution between two tubes of same size.

From the preceding definition, the t-Product can be seen as a generalization of the matrix product between \( n_1 \times n_2 \) and \( n_2 \times n_4 \) matrices by replacing the scalar to scalar multiplication (i.e. the \( \cdot \) in \( C_{ij} = \sum_{k=1}^{n_2} A(i,k) \cdot B(k,j) \)) with fiber to fiber circulant convolution (i.e. the * in Eq.(2.4.29)). We then replicate four additional definitions based on the t-Product from [73] in the following.

**Definition 2.4.2. (Tensor Conjugate Transpose [73])** The conjugate transpose of a tensor \( A \) of size \( n_1 \times n_2 \times n_3 \) is the \( n_2 \times n_1 \times n_3 \) tensor \( A^\top \) obtained by conjugate transposing each of the frontal slice and then reversing the order of transposed frontal slices 2 through \( n_3 \).

**Definition 2.4.3. (Identity Tensor [73])** A tensor \( J \in \mathbb{R}^{n \times n \times n_3} \) is called identity tensor if its first frontal slice \( J^{(1)} \) is the \( n \times n \) identity matrix and all its other frontal slices, i.e. \( J^{(i)} \) for \( i = 2, ..., n_3 \), are zero matrices.

**Definition 2.4.4. (Orthogonal Tensor [73])** A tensor \( Q \in \mathbb{R}^{n \times n \times n_3} \) is called orthogonal if the following condition holds,

\[
Q^\top * Q = Q * Q^\top = J,
\]

where \( J \in \mathbb{R}^{n \times n \times n_3} \) is an identity tensor as in Definition 2.4.3 and * is the t-Product as in Definition 2.4.1.

**Definition 2.4.5. (f-Diagonal Tensor [73])** For a tensor \( A \), if all its frontal slices \( A^{(i)} \), \( i = 1, ..., n_3 \) are diagonal matrices, then it is defined to be an f-diagonal tensor.

Finally, the t-SVD definition is formalized in Definition 2.4.6 and Fig. 2.1 gives an illustration of the t-SVD on an \( n_1 \times n_2 \times n_3 \) tensor.

**Definition 2.4.6. (Tensor Singular Value Decomposition (t-SVD) [73])** For \( A \in \mathbb{R}^{n_1 \times n_2 \times n_3} \), the t-SVD of \( A \) is given by

\[
A = U * S * V^\top,
\]

(2.4.31)
where \( U \in \mathbb{R}^{n_1 \times n_1 \times n_3} \) and \( V \in \mathbb{R}^{n_2 \times n_2 \times n_3} \) are orthogonal tensors as in Definition 2.4.4. \( S \in \mathbb{R}^{n_1 \times n_2 \times n_3} \) is a f-diagonal tensor as in Definition 2.4.5, whose entries are called singular tubes of \( \mathcal{A} \). Recall that the \( \ast \) here is the t-product as in Definition 2.4.1.

**Algorithm 2 t-SVD: \((U, S, V) = \text{tsvd}(\mathcal{A})\)**

**Input:** \( \mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3} \)

\[
\hat{\mathcal{A}} = \text{fft}(\mathcal{A}, [], i);
\]

for \( i = 1, 2, \ldots, n_3 \) do

\[
[U, S, V] = \text{svd}(\hat{\mathcal{A}}^{(i)})
\]

\[
\hat{U}^{(i)} = U; \hat{S}^{(i)} = S; \hat{V}^{(i)} = V;
\]

end for

\[
U = \text{ifft}(\hat{U}, [], i); \quad S = \text{ifft}(\hat{S}, [], i);
\]

\[
V = \text{ifft}(\hat{V}, [], i);
\]

**Output:** \( U, S, V \)

Considering the equivalence between the t-Production (essentially circulant convolution) in the original domain and the matrices multiplication in the Fourier domain, it is more convenient to carry out the t-SVD related computation in the Fourier domain. For a tensor \( \mathcal{A} \), the following introduces the notation for the Fourier transformed tensor \( \hat{\mathcal{A}} \) and a block diagonal matrix organized from \( \hat{\mathcal{A}} \).

**Definition 2.4.7. (Block Diagonal Matrix of Third Order Tensor [154])**

For a third order tensor \( \mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3} \), let \( \hat{\mathcal{A}} \) denote the Discrete Fourier trans-
formation (DFT) of $\mathcal{A}$, which can be computed by Matlab command $\text{fft}$ as $
abla \mathcal{A} = \text{fft}([\cdot],[\cdot],3)$. Define the block diagonal operation by $\text{blockdiag}$ and denote the computed block diagonal matrix by $\hat{\mathcal{A}}$, which are detailed as follows,

$$
\hat{\mathcal{A}} := \text{blockdiag}(\mathcal{A})
$$

$$
\begin{bmatrix}
\hat{\mathcal{A}}^{(1)} \\
\hat{\mathcal{A}}^{(2)} \\
\vdots \\
\hat{\mathcal{A}}^{(n_3)}
\end{bmatrix}
\in \mathbb{C}^{n_1 n_2 \times n_2 n_3}.
\tag{2.4.32}
$$

Algorithm 2 shows the algorithm for computing the t-SVD of $\mathcal{A}$, which is mainly based on the matrix SVD of $\hat{\mathcal{A}}^{(1)}$ to $\hat{\mathcal{A}}^{(n_3)}$ in Eq.(2.4.32). In addition, spectral norm and Frobenius norm can be extended to tensor and are related to the matrix norms of $\hat{\mathcal{A}}$, as detailed in the next two definitions.

**Definition 2.4.8. (Tensor Spectral Norm [153])** For a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the tensor spectral norm $\| \mathcal{A} \|_2$ is defined to be the spectral norm of $\hat{\mathcal{A}}$, i.e. $\| \mathcal{A} \|_2 := \| \hat{\mathcal{A}} \|_2$.

**Definition 2.4.9. (Tensor Frobenius Norm [153])** For a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the tensor Frobenius norm is denoted by $\| \mathcal{A} \|_F$, i.e. $\| \mathcal{A} \|_F := (\langle \mathcal{A}, \mathcal{A} \rangle)^{1/2} = \frac{1}{\sqrt{n_3}} \| \hat{\mathcal{A}} \|_F = \frac{1}{\sqrt{n_3}} \| \hat{\mathcal{A}} \|_F = \sqrt{\sum_i \sum_j \sum_k \mathcal{A}_{ijk}^2}$.

### 2.4.3 Tensor Tubal Rank Definition and Tensor Nuclear Norm

T-SVD arises the following tensor tubal rank definitions.

**Definition 2.4.10. (Tensor tubal multi-rank, tubal rank and average of tubal multi-rank, sum of tubal multi-rank)** For a third order tensor $\mathcal{A}$, let $\hat{\mathcal{A}} = \text{fft}([\cdot],[\cdot],3)$ and $\hat{\mathcal{A}} = \text{blockdiag}(\hat{\mathcal{A}}) = \text{blockdiag}(\hat{\mathcal{A}}^{(1)}, \ldots, \hat{\mathcal{A}}^{(n_3)})$. Let the t-SVD be $\mathcal{A} = U \ast S \ast V^T$. Define the following terms:

- **Tubal multi-rank**: $r(\mathcal{A}) := (r_1, \ldots, r_i, \ldots r_{n_3})$, where $r_i = \text{rank}(\hat{\mathcal{A}}^{(i)})$;

- **Tubal rank**: $\text{rank}_i(\mathcal{A}) := \# \{ i : S(i, :, :) \neq 0 \} = \max_i r_i$;
- **Average of tubal multi-rank:** \( \text{rank}_{\text{avg}}(\mathcal{A}) := \frac{1}{n_3} \sum_{i=1}^{n_3} r_i; \)

- **Sum of tubal multi-rank:** \( \text{rank}_{\text{sum}}(\mathcal{A}) := \sum_{i=1}^{n_3} r_i. \)

The tensor nuclear norm (TNN) seeks a convex surrogate to the tensor tubal-related rank. There are two existing definitions of TNN, which are based on different convex relaxation and result into different definitions of TNN.

**Definition 2.4.11. (Averaged Tensor Nuclear Norm [88, 64])** For a tensor \( \mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3} \), the averaged tensor nuclear norm \( \| \mathcal{A} \|_{t^*} \) is defined to be the average of nuclear norm of all the frontal slices of \( \hat{\mathcal{A}} \),

\[
\| \mathcal{A} \|_{t^*, \text{avg}} = \frac{1}{n_3} \sum_{i=1}^{n_3} \| \hat{\mathcal{A}}^{(i)} \|_*.
\] (2.4.33)

**Definition 2.4.12. (Summed Tensor Nuclear Norm [118, 154, 153])** For a tensor \( \mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3} \), the summed tensor nuclear norm \( \| \mathcal{A} \|_{t^*} \) is defined to be the sum of nuclear norm of all the frontal slices of \( \hat{\mathcal{A}} \),

\[
\| \mathcal{A} \|_{t^*, \text{sum}} = \sum_{i=1}^{n_3} \| \hat{\mathcal{A}}^{(i)} \|_*.
\] (2.4.34)

The averaged TNN is based on average of tubal multi-rank, while the summed TNN is based on tubal multi-rank. They differ with a \( \frac{1}{n_3} \) factor, which is important because without it, TNN and the tensor spectral norm are not dual to each other, which leads to inconsistency when reduced to the matrix case.

### 2.5 Differentially Private Learning

The formal definition of differential privacy for a randomized algorithm \( \mathcal{ALG} \) with parameter \( \epsilon \) and \( \delta \) is as follows.

**Definition 2.5.1. ((\( \epsilon, \delta \))-Differential Privacy ((\( \epsilon, \delta \))-DP))** A randomized algorithm \( \mathcal{ALG} \) is \((\epsilon, \delta)-\)differentially private if, for all neighboring data sets \( \mathcal{D} \) and \( \mathcal{D}' \), which differ in only one data sample and for all outputs \( \mathcal{O} \) we have \( \Pr(\mathcal{ALG}(\mathcal{D}) \in \mathcal{O}) \leq e^\epsilon \Pr(\mathcal{ALG}(\mathcal{D}') \in \mathcal{O}) + \delta \).
2.5.1 Differentially Private Optimization for Feature-wise Distributed Dataset

Feature-wise distributed private learning

Feature-wise distributed data is more challenging than sample-wise distributed dataset under privacy restriction. For the latter setting, each user node has enough information to take local update (e.g. user can compute the local gradient based on local data samples) and only the decision variables are needed to be communicated. However, for feature-wise distributed data, apart from the decision variable, additional information is required to be shared to perform local update (e.g. compute local partial block-wise gradient). In general, more information sent by the user node, more likely sensitive individual privacy is at risk, which makes the privacy protection design more challenging. As a largely unexplored setting, to the best of our knowledge, the very recent [53] is the only exiting work that has considered the same differentially private ERM learning task with disjoint features held by different parties. They propose to add privacy protection during preprocessing by communicating perturbed sketched features [70]. Although the uplink communication is one-shot during the preprocessing and its sketching step partially relieves the high communication complexity in terms of the feature dimension $d$, its complexity is still linearly dependent on the sample size $n$ (i.e. $O(n)$). In comparison, our method only communicates active features indicated by the optimization procedure, featuring a “share-at-need” strategy. As a result, to achieve the nearly optimal utility, the overall uplink complexity of our method is $O(n^{2/3} \log(n^{1/3}))$, which is more uplink communication efficient.

Private conditional gradient algorithm

[129] proposes a centralized private conditional gradient algorithm for ERM problem constrained by atomic norm. In each iteration, the FW algorithm greedily selects a linear oracle from the atomic norm set $\mathcal{A}$ (has finite number of atomic norm)
by picking the one with the largest duality gap. \cite{129} selects the iterative linear oracle by Report-Noisy-Max mechanism \cite{35} (a special variant of the more general exponential mechanism), which ensures the differential privacy. For the LASSO task, \cite{129} is proved to provide nearly optimal utility guarantee. Since the utility guarantee is based on the convergence analysis, the adaptation of the method to distributed setting is non-trivial due to the missing convergence result for BCFW-AS \cite{3}. Furthermore, with features distributed among user nodes, apart from the linear oracle evaluation, the gradient computation also requires additional perturbation for privacy protection, whose effect on utility demands careful quantization and further analysis.

### 2.5.2 Differentially Private Streaming Convex Optimization

Given a streaming sequence of loss functions $I = [f_1, f_2, ..., f_t, ... f_T]$ arriving one at a time, the COCO algorithm is required to response $x_t$ from the constraint set $C$, which is a bounded convex set. After each response, it will suffer a convex loss $f_t(x_t)$. In machine learning, depending on the task, the function $f_t$ can have various choices, for example logistic loss or hinge loss for classification, square function for linear regression. We measure the utility by regret, a common notion used in online algorithms, defined as:

**Definition 2.5.2.** (Regret) Denote the private release of the algorithm by $x_1, x_2, ..., x_t, ..., x_T$, then the regret with sequence length $T$ is $\text{Regret}(T) = \sum_{t=1}^{T} f_t(x_t) - \min_{x \in C} \sum_{t=1}^{T} f_t(x)$.

It is common to consider a relaxed differential privacy in the streaming setting, which is based on the simple yet practical privacy expiration assumption. In detail, the window differential privacy definition only delivers privacy protection for recent instances inside a sliding window \cite{9}. That is, only the changes of output caused by the latest $W$ individual entries are counted into the privacy loss, while the changes

---

\footnote{\textsuperscript{3}As most existing conditional gradient methods are referred by Frank-Wolfe algorithm in the coordinate descent variants, we will use the term Frank-Wolfe algorithm for coordinate descent variants of conditional gradient algorithms}
of the output caused from distant inputs are not concerned. The following definition formalizes the window differential privacy, which is adapted from [9] to our COCO setting.

**Definition 2.5.3. (Window Differential Privacy (WDP))** Let $\mathcal{A}$ be a random algorithm. With a particular sequence length $T$, for any input convex function sequences $\mathcal{I}$ and a neighboring sequence $\mathcal{I'}$, the output sequence space $\mathcal{O}$, $\mathcal{A}$ is window $(\epsilon, \delta)$-differential privacy with the window size $W$ if the following condition holds,

$$P[\mathcal{A}([f_1, f_2, ..., f_t, ..., f_T]) \in \mathcal{O}] \leq e^{w(T-t)\epsilon} P[\mathcal{A}([f_1, f_2, ..., f'_t, ..., f_T]) \in \mathcal{O}] + \delta,$$

(2.5.35)

where $w(i) = 1$ for $i < W$, $w(i) = \infty$ for $i \geq W$. We say the algorithm is window $\epsilon$-differentially private with $W$ if $\delta = 0$. 


Chapter 3

Efficient Generalized Conditional Gradient with Gradient Sliding for Composite Optimization

3.1 Introduction

This chapter studies unconstrained composite optimization problems of the form

$$\min_{x \in \mathcal{X}} F(x) = l(x) + r(x), \quad (3.1.1)$$

where $\mathcal{X}$ is a general vector space equipped with inner product (e.g. $\mathbb{R}^d$ equipped with $\ell_2$ norm). $l(x)$ is a smooth convex loss function, i.e. it is continuously differentiable with $L$-Lipschitz continuous gradient:

$$||\nabla l(x) - \nabla l(y)|| \leq \frac{L}{2} ||x - y||, \quad \forall x, y \in \mathcal{X}. \quad (3.1.2)$$

$r(x)$ stands for the regularizer which is a nonsmooth closed proper convex function. Problem (3.1.1) is of vital importance in machine learning because many sparse estimation problems fit into this model. For example, in convex relaxed low rank matrix completion problem [17], $l(x)$ is the Frobenius norm between the observed matrix and the low rank estimation matrix, and $r(x)$ is the trace norm of the low rank estimation matrix. In regularized empirical risk minimization problem, $l(x)$ can
be logistic loss or least square loss, and $r(x)$ can vary from simple ones like $\ell_1$ norm to very complex form such as graph-guided lasso and group lasso for inducing structured sparsity.

Many different lines of methods exist for solving such sparse optimization problems, among them is the popular proximal gradient (PG) based approach. This kind of methods can achieve the optimal rate of convergence under certain problem settings, hence it enjoys low iteration complexity. The per-iteration cost mainly comes from gradient evaluation and a proximal map (PM) related to the type of the regularizer. On the one hand, due to the optimal iteration complexity, the number of gradient evaluations is optimal for PG method. On the other hand, the proximal map itself is a quadratic optimization problem composed by $r(x)$ and a quadratic term. For particular regularizers, it admits efficient evaluation. For example, the proximal map related to the lasso regularizer is simply the soft-thresholding. However, in some cases, evaluating the proximal map can be quite computational demanding. For instance, to solve the proximal map related the trace norm regularizer, it requires a full singular value decomposition (SVD) in each iteration. As a result, the high per-iteration cost raised by the proximal map becomes the bottleneck of the PG method for those problems.

To address the high per-iteration cost raised by the PM, generalized conditional gradient method (GCG) has been receiving increasing research interest. It only requires to evaluate a linear operator (LO) in each iteration, which is intuitively much easier to solve than the quadratic subproblem of PM. In fact, this intuition is elaborated by many structured sparse regularizer, e.g. LO requiring spectral norm versus PM requiring full SVD for trace norm regularized problem, in which the former is much more computational efficient. As a result, although the iteration complexity of GCG based methods are inferior to their PG counterpart, some studies have found that the low per-iteration cost can sometimes compensate for the extra iterations leading to less overall execution time than PG. Nevertheless, the inferior iteration complexity leads to extra gradient
evaluations. Although the per-iteration cost for LO can be small enough to afford excess iterations, the increased demand for gradient evaluation can raise an inevitable trade-off, especially for large scale problems. In addition, the convergence results become weaker when approximation techniques are introduced to solve LO. As a result, these approximation techniques also bring about an increased count of gradient evaluations, which can counteract the efficiency gained from the accelerated LO evaluations.

In this chapter, we therefore propose a novel algorithm called Generalized Conditional Gradient with Gradient Sliding (GCG-GS) and its refined variant for gauge regularized problems. We first extend a recent optimization scheme called gradient sliding to general unconstrained composite convex optimization problems. Instead of evaluating gradient on each iteration, we skip it from time to time, which can be viewed as many LO evaluations sharing the same gradient value. For gauge regularized problem ($60$; $144$), where efficient approximation techniques exist for handling $r(x)$, we present an improved variant of the general GCG-GS algorithm to incorporate these techniques to further accelerate the algorithm. As a result, our algorithm has optimal count of gradient evaluations as their PG counterpart, and more importantly, it allows efficient approximation techniques to be used without increasing the optimal count of gradient evaluation. Experiments of CUR-like matrix factorization problem with group lasso penalty on four real-world datasets have demonstrated the efficiency of the proposed method.

3.2 Preliminary

3.2.1 Conditional Gradient Sliding Algorithm (CGS)

Paper $31$ has proposed a gradient sliding technique for constraint smooth objective function to reduce gradient evaluations of CG algorithm. Under CG framework, although the requirement for solving linear operator is still $O(\frac{1}{\epsilon})$, gradient evaluations can be surprisingly reduced to match the iteration complexity of PG counterpart
under the same problem settings.

The CGS separates into outer and inner iteration. On each outer iteration, two additional sequences are maintained. It can be seen as a variant to Nesterov’s optimal gradient method [101], with the modification of the step 3.2.5 calling a CG subroutine rather than a gradient descent or proximal mapping procedure for PG, as shown in the following:

\[ z_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_k; \quad (3.2.3) \]
\[ g_k = \nabla F(z_{k+1}); \quad (3.2.4) \]
\[ x_{k+1} = CG(g_k, x_k, \beta_k, \eta_k); \quad (3.2.5) \]
\[ y_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_{k+1}. \quad (3.2.6) \]

The inner loop, namely the CG subroutine, applies the classic CG algorithm to optimize the following subproblem:

\[ \phi(v) = \langle v, g_k \rangle + \frac{\beta_k}{2} ||v - x_k||^2. \quad (3.2.7) \]

According to CG method, on each inner iteration, it optimizes the duality gap:

\[ v_t = \arg \max_{v \in D} G(u_t, v) = \arg \max_{v \in D} \langle u_t - v, \nabla \phi(u_t) \rangle, \quad (3.2.8) \]

where \( u_t \) is the solution sequence of the inner loop subroutine\(^1\) and \( D \) is the constraint set of the problem [81] considers. Here, we have slightly extended the notation of the duality gap to incorporate the additional variable \( v \). The subprocess returns the latest \( u_t \) once the duality gap is less than \( \eta_k \). That is, it returns \( u_{tk} \) when

\[ G(u_{tk}, v_{tk}) \leq \eta_k. \quad (3.2.9) \]

As a result, it can be viewed as if many LO evaluations can share the same gradient to maintain the same convergence rate, instead of updating the gradient for each LO evaluation. Algorithm 3 is the conditional gradient sliding algorithm. In step 3 it evaluates the gradient of \( F(x) \) at \( z_{k+1} \). It can be viewed as a variant of Nesterov’s accelerated gradient method, with step 4 changed to a sub-routine call CG instead

\(^1\)We use \( k \) to represent sequences related to outer loop and \( t \) for inner loop in this chapter.
of directly evaluating the linear operator using $g_k$. Algorithm 4 is the CG subroutine. It applies the classic CG algorithm on $\phi(x)$ until its duality gap is smaller than the predefined $\eta$.

**Algorithm 3** Conditional Gradient Sliding (CGS) algorithm

**Input:** $x_0, K, \gamma_k, \beta_k, \eta_k$

1: for $k = 0, 1, \ldots, K - 1$ do
2: \begin{align*}
    z_{k+1} &= (1 - \gamma_k) y_k + \gamma_k x_k; \\
    g_k &= \nabla F(z_{k+1}); \\
    x_{k+1} &= CG(g_k, x_k, \beta_k, \eta_k); \\
    y_{k+1} &= (1 - \gamma_k) y_k + \gamma_k x_{k+1}; \\
\end{align*}
6: end for

**Output:** $y_K$

**Algorithm 4** CGS subroutine for inner iteration

**Input:** $g, u, \beta, \eta$

1: let $\phi(x) = \langle g, x \rangle + \frac{\beta}{2} ||x - u||^2$, $u_0 = u; \quad$ 2: for $t = 0, 1, \ldots,$ do
3: \begin{align*}
    G(u_t, v) &= \langle u_t - v, \nabla \phi(u_t) \rangle; \\
    v_t &= \arg \max_{v \in \mathcal{D}} G(u_t, v); \\
    \text{if } G(u_t, v_t) \leq \eta \text{ then} \quad$ 6: break; \quad$\text{end if} \quad$ 7: $\alpha_t = \arg \min_{\alpha \in [0, 1]} \phi((1 - \alpha)u_t + \alpha v_t); \quad$ 8: $u_{t+1} = (1 - \alpha_t)u_t + \alpha_t v_t; \quad$ 9: \end{align*}
10: return $u^+ = u_t; \quad$

Moreover, with $F(x)$ not only smooth but also strongly convex, calls CGS algorithm stage-wisely, where the output of the last stage is used to warm start the current stage. In this scheme, the precision $\eta_k$ of inner subroutine is also modified to be not only related to $k$ but also to $s$. Algorithm details this restart scheme.
Algorithm 5 Restart CGS for Strongly Convex Problem

Input: $x_0$

1: for $s = 0, 1, ..., do$
2: \hspace{1em} p_{s+1} = \text{CGS}(p_s, K, \eta_s);$
3: end for

Output: $p_s$

3.3 The Proposed Algorithm

In this section, we propose our novel algorithm for Problem (3.1.1), called Generalized Conditional Gradient with Gradient Sliding (GCG-GS). We first present the GCG-GS for general regularizer. Our algorithm is related to [81], but it can suit to more general composite optimization problems with the unconstrained domain. Although for some problems it can be equivalently transformed between regularization and constraint form by Lagrangian duality, the regularization form we consider here allows additional heuristic local optimization, which is hardly known for constraint form. More importantly, we will propose a refined GCG-GS algorithm, which admits various approximation techniques for further accelerating the LO evaluation. When the approximation techniques are involved, neither [81] nor our general GCG-GS are applicable. This issue is mainly because the stopping criteria of Eq.(3.2.9) is either no longer computable, or computationally expensive to obtain. Our refined GCG-GS will handle this issue.

3.3.1 General GCG-GS

Algorithm Description:

To involve the gradient sliding scheme, our algorithm also separates into outer and inner loops. In the outer loop, we evaluate the gradient of the smooth part $g_k = \nabla l(z_{k+1})$. Then the subroutine is called. As for the subroutine, we consider $\Phi(v)$ composed of $\phi(v) = \langle v, g \rangle + \frac{\beta}{2}||v - u||^2$ and nonsmooth $r(v)$, i.e. $\Phi(v) = \phi(v) + r(v)$. 

33
We use a different definition of $G(u_t, v)$ according to Eq. (2.1.3), which is to be optimized as follows:

$$G(u_t, v_t) = \max_{v \in \mathcal{X}} \{ r(u_t) - r(v) - \langle v - u_t, \nabla \phi(u_t) \rangle \}. \quad (3.3.10)$$

Note that $G(u_t, v_t)$ is the duality gap of $\Phi(v)$ at $u_t$. Thus, the subroutine actually solves the minimization problem of $\Phi(v)$ by GCG algorithm until certain duality gap is obtained. In addition, $\alpha_t$ can be optimally chosen by solving

$$\alpha_t = \arg \min_{\alpha \in [0,1]} \phi((1 - \alpha)u_t + \alpha v_t) + r((1 - \alpha)u_t + \alpha v_t). \quad (3.3.11)$$

Finally, the next variable is obtained by

$$u_{t+1} = (1 - \alpha_t)u_t + \alpha_t v_t. \quad (3.3.12)$$

For clarity, we summarize the general GCG-GS in Algorithm (6) and Algorithm (7).

**Algorithm 6** General-GCG-GS

**Input:** $x_0, K, \gamma_k, \beta_k, \eta_k$

1: for $k = 0, 1, \ldots, K - 1$ do

2: \quad $z_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_k$;

3: \quad $g_k = \nabla l(z_{k+1})$;

4: \quad $x_{k+1} = GCG(g_k, x_k, \beta_k, \eta_k)$;

5: \quad $y_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_{k+1}$;

6: end for

**Output:** $y_K$
Algorithm 7 GCG: General-GCG-GS subroutine

Input: $g, u, \beta, \eta$

1: let $\phi(x) = \langle g, x \rangle + \frac{\beta}{2}||x - u||^2$, $u_0 = u$

2: for $t = 0, 1, \ldots$, do

3: $G(u_t, v) = r(u_t) - r(v) - \langle v - u_t, \nabla \phi(u_t) \rangle$

4: $v_t = \arg \max_{v \in \mathcal{X}} G(u_t, v)$

5: if $G(u_t, v_t) \leq \eta$ then

6: break

7: end if

8: $\alpha_t = \arg \min_{\alpha \in [0, 1]} \phi((1 - \alpha)u_t + \alpha v_t) + r((1 - \alpha)u_t + \alpha v_t)$

9: $u_{t+1} = (1 - \alpha_t)u_t + \alpha_t v_t$

10: end for

11: Return: $u^+ = u_t$

Convergence Analysis:

For the general GCG-GS algorithm, we have the following convergence guarantee.

We first introduce the following notation:

$$\Gamma_0 = 1; \quad \Gamma_k = \Pi_{i=1}^{k} (1 - \gamma_i), k = 1, 2, \ldots$$ (3.3.13)

**Theorem 3.3.1.** Under finite solution sequence assumptions, apply GCG-GS to Problem (3.1.1),

a) for any $x$, the output $y_K$ satisfies,

$$F(y_K) - F(x) \leq \Gamma_{K-1} (1 - \gamma_0) (F(y_0) - F(x)) + \sum_{k=0}^{K-1} \frac{\Gamma_{K-1} \gamma_k \beta_k}{2\Gamma_k} (||x_k - x||^2 - ||x_{k+1} - x||^2)$$ (3.3.14)

$$+ \sum_{k=0}^{K-1} \frac{\Gamma_{K-1} \gamma_k \eta_k}{\Gamma_k}$$

b) consider the inner loop, namely the LO evaluations, for a particular stage $k$, denote $\beta = \beta_k$, the duality gap satisfies,

$$\min_{i=0}^{t} G(u_i, v_i) \leq \frac{6\beta D_s}{t + 2}$$ (3.3.15)
where \( D_s \) is the upper bound of the solution sequence.

**Corollary 3.3.2.** With the sequences setting as \( \beta_k = \frac{2L}{k+1}, \gamma_k = \frac{2}{k+2}, \eta_k = \frac{2LD_0}{K(k+1)} \) and denoting \( D_0 = ||x_0 - x||^2 \), we have:

a)

\[
F(y_K) - F(x) \leq \frac{6LD_0}{K(K+1)}.
\]

(3.3.16)

For finding an \( \epsilon \) solution, we get:

\[
K = \sqrt{\frac{6LD_0}{\epsilon}}.
\]

(3.3.17)

b) the total number of inner LO evaluations is

\[
T_K = \frac{6D_s}{D_0}K^2 + K.
\]

(3.3.18)

For finding an \( \epsilon \) solution, we have:

\[
T_K = \frac{36LD_s}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}.
\]

(3.3.19)

The proof can be found in the Appendix A.

**Discussion:**

There are various assignment of sequence, please see \([81]\). For the particular sequence we adopt here, it is apparent that our number of gradient evaluations for finding an \( \epsilon \) solution is \( O(\sqrt{\frac{1}{\epsilon}}) \), which is the same as those optimal complexity of PG methods for Problem (3.1.1). In addition, in terms of the total number of LO evaluations \( T_K \), the proposed method maintains the same order of complexity as those plain GCG methods which is also optimal for GCG.

However, the above algorithm is only conceptual in some sense. Note that both the Subproblems (3.3.10) and (3.3.11) can be difficult to solve for some \( r(x) \). Also, even they were solvable, we often prefer avoiding directly computing \( r(x) \) by considering more efficient substitution such as polar operator in Section 2.1.3. Inspired by this, we will propose the refined GCG-GS algorithm in the next subsection to allow more efficient inner loop execution.
3.3.2 Refined GCG-GS for Gauge Regularized Problem

In this subsection, we follow the assumption as in Section 2.1.3, where the regularizer is a generalized gauge function defined by Eq. (2.1.7). Note that most practically used sparsity inducing and rank minimization regularizers can be seen as generalized gauge function.

Efficient Approximation Techniques:

Essentially, we apply the efficient approximation techniques introduced in Section 2.1.3 to minimize \( \Phi(v) \). To efficiently minimize Eq. (3.3.10), we first convert it to constraint form:

\[
v_t \in \arg \min_{v: h(v) \leq \zeta} \langle v, \nabla \phi(u_t) \rangle. \tag{3.3.20}
\]

Then we update \( v_t \) by solving the direction and scalar separately, namely \( \alpha_t v_t \approx \theta_t a_t \) (\( \theta_t \) denotes the approximate scalar). The direction is updated by

\[
a_t \in \arg \min_{a: \kappa(a) \leq 1} \langle a, \nabla \phi(u_t) \rangle. \tag{3.3.21}
\]

Incorporating the approximation Eq. (2.1.12) and solving it on atomic domain \( \mathcal{A} \), we can obtain \( a_t \) more efficiently by

\[
\langle a_t, \nabla \phi(u_t) \rangle \leq \epsilon_t + \min_{a \in \mathcal{A}} \langle a, \nabla \phi(u_t) \rangle = \epsilon_t - \kappa^0(-\nabla \phi(u_t)). \tag{3.3.22}
\]

The scalar, denoted as \( \theta_t \), would be originally chosen as

\[
\theta_t = \arg \min_{\theta} \phi((1 - \alpha_t)u_t + \theta a_t) + h((1 - \alpha_t)u_t + \theta a_t)), \tag{3.3.23}
\]

where \( \alpha_t \) is a deterministic sequence to be specified in our next theorem. Again, to avoid direct evaluation of \( \kappa(u_t) \), an upper substitution \( \rho_t \) is used as in [144]. This is achieved by choosing \( \rho_0 \geq \kappa(u_0) \) and the update scheme \( \rho_{t+1} = (1 - \alpha_t)\rho_t + \theta_t \). Then \( \rho_t \geq \kappa(u_t) \) can be held iteratively, see [144]. Thus, by alternatively using \( h(\rho_t) \) provided that \( h(\cdot) \) is increasing convex and \( \kappa(\cdot) \) is convex, \( \theta_t \) can be obtained by

\[
\theta_t = \arg \min_{\theta} \phi((1 - \alpha_t)u_t + \theta a_t) + (1 - \alpha_t)h(\rho_t) + \alpha_t h\left(\frac{\theta}{\alpha_t}\right). \tag{3.3.24}
\]
Finally, an additional local heuristic optimization can be adopted to further improve the practical performance, which is another motivation for using the regularized form rather than constrained form. Denoting such re-optimization by \textbf{Improve}, we adopt the following conceptual requirement for it, which is \textbf{Relaxed} assumption according to [144]:

\[
\begin{aligned}
&\phi(u_{t+1}) + h(\rho_{t+1}) \leq \phi(u_t) + \langle \tilde{u}_{t+1} - u_t, \nabla \phi(u_t) \rangle \\
&\quad + \frac{\beta}{2} ||\tilde{u}_{t+1} - u_t||^2 + (1 - \alpha_t) h(\rho_t) + \alpha_t h\left(\frac{\theta_t}{\alpha_t}\right) ; \\
&\rho_{t+1} \geq \kappa(u_{t+1}).
\end{aligned}
\]  
\quad (3.3.25)

\textbf{Weighted Average as Return Value:}

An important issue with the above approximation is that the duality gap Eq.(3.3.10) is either no longer computable, or even when we can compute it, it is unreasonable for us to directly evaluate it because we do all the above approximations to avoid computing $\kappa(v)$ directly. As a result, the stopping criteria in general GCG-GS algorithm (also CGS algorithm) cannot be used. Furthermore, the choice of return value becomes a problem because the previous bound on duality gap only guarantees the minimum one. Again, as we cannot directly compute the duality gap, it also becomes unknown that on which particular $u_t$ the duality gap is small enough.

To solve the above stopping criteria problem, we propose a simple alternative by estimating a maximum iteration count $m$ of the inner LO evaluation loop. As shown in our convergence analysis, different outer loops can share the same $m$.

As for the choice of return value, instead of returning a particular $u_t$, we propose using the weighted average of $u_t$ as the returned value. We show such averaged $\bar{u}_m$ can guarantee $G(\bar{u}_m, x)$ to be smaller than the desired $\eta$ as long as the proper approximated $m$ is used. In detail, the return value to the outer loop is

\[
\bar{u}_m = \sum_{t=0}^{m-1} \nu_t u_t, \quad \nu_t = \frac{2}{m(m + 1)} (t + 1).
\]  
\quad (3.3.26)

Intuitively, variables of later iterations gain more weights. This intuition is compatible to the analysis in [59], where the one achieving the smallest duality gap lies in the
last third iterations. This average scheme is by observing the special construct of \( \phi \), namely, the corresponding \( G(u_t, x) \) is convex in \( u_t \) for arbitrary yet fixed \( x \). Note that it also allows an online update \( \bar{u}_{t+1} = (1 - \frac{2}{t+2})\bar{u}_t + \frac{2}{t+2}u_t \), which is exactly what has been shown in the algorithm. We note that [77] has used the same weighted average as an update option in block coordinate conditional gradient method. To summarize together, Algorithm (8) and Algorithm (9) show the implementation details of the Refined-GCG-GS.

**Algorithm 8** Refined-GCG-GS

**Input:** \( x_0, m, \beta_k, \gamma_k \)

1: for \( k = 0, 1, ..., K - 1 \) do
2: \( z_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_k \);
3: \( g_k = \nabla l(z_{k+1}) \);
4: \( x_{k+1} = \text{Re-GCG}(g_k, x_k, \beta_k, m) \);
5: \( y_{k+1} = (1 - \gamma_k)y_k + \gamma_k x_{k+1} \);
6: end for

**Output:** \( y_K \)

**Algorithm 9** Re-GCG:Refined-GCG-GS subroutine

**Input:** Input from outer loop: \( g, u, \beta, m \); Sequence \( \alpha_t \)

1: let \( \phi(x) = \langle g, x \rangle + \frac{\beta}{2}||x - u||^2_2 \), \( u_0 = u \);
2: for \( t = 0, 1, ..., m - 1 \) do
3: choose \( a_t \) satisfy \( \langle a_t, \nabla \phi(u_t) \rangle \leq \epsilon_t - \kappa^\alpha(-\nabla \phi(u_t)) \);
4: \( \theta_t = \text{arg min}_\theta \phi((1 - \alpha_t)u_t + \theta a_t) + (1 - \alpha_t)h(\rho_t) + \alpha_t h(\frac{\theta}{a_t}) \);
5: \( \tilde{u}_{t+1} = (1 - \alpha_t)u_t + \theta_t a_t \);
6: \( \tilde{\rho}_{t+1} = (1 - \alpha_t)\rho_t + \theta_t \);
7: \( (u_{t+1}, \rho_{t+1}) = \text{Improve}(\tilde{u}_{t+1}, \tilde{\rho}_{t+1}, \phi, r) \);
8: \( \bar{u}_{t+1} = (1 - \frac{2}{t+2})\bar{u}_t + \frac{2}{t+2}u_{t+1} \);
9: end for
10: Return: \( u^+ = \bar{u}_m \);

39
Convergence Analysis:

**Theorem 3.3.3.** Let the sequence settings be $\beta_k = \frac{2H}{k+1}$, $\gamma_k = \frac{2}{k+2}$, $\alpha_t = \frac{2}{t+2}$ and the inner loop count $m = \left\lceil \frac{6K(D_s + \delta \kappa(x))}{D_0} \right\rceil$, where $D_s$ is an upper bound on the distance of the solution path, $D_0$ is the distance between $x_0$ and $x$, $\delta$ is the constant satisfies $\epsilon_i \leq \frac{\delta \beta_k \alpha_t}{\epsilon}$. Then the following results hold for the above algorithm for any $x$ in $\mathcal{X}$.

a) After $K$ outer loops, the output $y_K$ satisfies

$$F(y_K) - F(x) \leq \frac{6LD_0}{K(K + 1)};$$  \hspace{1cm} (3.3.27)

for finding an $\epsilon$ solution, and the number of FO evaluation requires

$$K = \sqrt{\frac{6LD_0}{\epsilon}}.$$  \hspace{1cm} (3.3.28)

b) The total number of LO evaluation $T_K$ for finding an $\epsilon$ solution requires

$$T_K = \frac{36L(D_s + \delta \kappa(x))}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}.$$  \hspace{1cm} (3.3.29)

Proof can be found in Appendix.

**Discussion:**

The outer loop complexity in this subsection is exactly the same as the one in the previous subsection, despite all the approximation we make to efficiently evaluate LO. As a result, the count of gradient evaluation keeps unchanged. As a sharp comparison, the convergence rate of \cite{59} is degenerated by a factor of 2 when $d_k$ is evaluated approximately. Apparently, the approximations made to the LO lead to the increasing count of evaluations of gradient.

We point out that, by properly restarting GCG-GS, this algorithm can also obtain optimal count of gradient evaluation for strongly convex problem \cite{81}. In this chapter, we only discuss the problem under convex assumption due to space limitation. In fact, the extension to strongly convex case is straightforward in some sense.
3.4 Experiment

In this section, we demonstrate the efficiency of the proposed algorithm by a CUR-like matrix factorization task (91; 92) regularized by group lasso penalty. This experiment was conducted by using MATLAB on a laptop computer of Intel Core i7 2.7GHz processor with 8 GB RAM.

Experiment Setup:

We consider the following CUR-like matrix factorization problem (92).

\[
\min_X \frac{1}{2} \|D - DXD\|^2_F + \lambda (\sum_i \|X_i\|_\infty + \sum_j \|X_j\|_\infty),
\]

(3.4.30)

where \(D\) is the input data matrix, \(X_i\) and \(X_j\) denote the row vectors and column vectors, respectively, and \(\| \cdot \|_\infty\) is the max norm. We set \(\lambda = 5 \times 10^{-4}\) in our experiment. We utilized the following four real datasets as used in [144]: SRBCT, Brain_Tumor_2, 9_Tumor and Leukemia, which are of sizes 83 \(\times\) 2308, 50 \(\times\) 10367, 60 \(\times\) 5762, and 72 \(\times\) 11225, respectively. For comparison, we utilized GCG_TUM algorithm in [171]. For our GCG-GS algorithm, we implemented the outer loop routine, which then called the same polar operator of GCG_TUM for inner loop subroutine. Hence, the improved performance is gained purely from the gradient sliding scheme. We set our inner loop estimation \(m\) to 3 for all four datasets. Other input sequences were assigned exactly as the theoretical part. Note that we did not compare with PG based methods because they have already been shown to be less efficient than GCG_TUM in [144].

Result:

Figure 3.4 shows the experimental results, where three sets of plots are drawn: objective function value versus total running time, versus polar operator evaluation time only and versus gradient evaluation time only. We sampled every 30 iterations.


\(^3\)Download from http://users.cecs.anu.edu.au/~xzhang/GCG.
Figure 3.1: Objective function value versus total running time, polar operator time only, and gradient time only.

for GCG_TUM and every 10 outer loop iterations for GCG-GS. In general, our algorithm is much faster than GCG_TUM algorithm in terms of convergence speed, as illustrated by the curve of objective function value versus total running time. Also, our algorithm requires much less time on gradient evaluation to achieve certain decrease of objective function value on all four datasets. Finally, the time requirements for polar operator evaluation of our algorithm are similar to GCG_TUM on Brain Tumor 2 and SRBCT, superior than GCG_TUM on Leucamia and inferior than GCG_TUM on 9 Tumor.

Choice of Inner Iteration Count m:

In this subsection, we study the effect of different estimates of $m$. We run the GCG-GS algorithm with 400 outer loops on the four datasets, while vary the maximum inner iteration from 2 to 7. Figure 3.4 plots the objective function value versus
number of iterations. With the different inner iteration number, the algorithm actually converges at similar outer loop count. To be specific, all lines begin to converge around 50 numbers of outer iteration. Although the algorithm converges to different objective function value with different $m$, the difference is below 0.005. In addition, we observed that $m = 3$ always yields relatively superior performance. Hence, $m$ is not hard to tune practically.

### 3.5 Summary

In this chapter, we have proposed a new algorithm under GCG framework. Our algorithm has optimal count of gradient evaluations as those PG method, which is an order superior than plain GCG methods. Also, it admits the incorporation of efficient approximation techniques for accelerating the evaluation of linear operator that CGS lacks. Meanwhile, this count of gradient requirement remains unchanged.
Experiment on a CUR-like matrix factorization task with group lasso penalty on four real datasets have demonstrated the efficiency of the proposed method.
Chapter 4

Proximal Average Approximated Incremental Gradient Method for Composite Penalty Regularized Empirical Risk Minimization

4.1 Introduction

Empirical risk minimization (ERM) is a fundamental machine learning method that learns the model by minimizing the average loss taken from the training data. To induce better prediction performance and introduce prior knowledge about the model, the empirical loss is often regularized by a penalty function. Based on the specific task, the penalty functions can vary from smooth functions like $\ell_2$-norm to nonsmooth simple functions like $\ell_1$-norm. Composite nonsmooth functions, recognized their ability to inducing structured sparsity model, have been intensively applied in bioinformatics and text mining tasks. However, it is difficult to efficiently optimize such composite penalty regularized ERM problems, especially when confronted with very large datasets.

In general, nonsmooth composite penalties, like overlapping group lasso or graph-
guided lasso, are hard to deal with. One fact is that the proximal gradient method \cite{7,103}, which is an effective approach to simple nonsmooth penalties, is not applied in this case because its crucial proximal mapping step is difficult to solve. That is, existing simple methods cannot be directly applied when engaging with these complex structured penalties. A splitting method called alternating direction method of multipliers (ADMM) \cite{13}, with its variants like stochastic ADMM and incremental ADMM with better scalability, has been extensively studied. Stochastic ADMM methods \cite{108} utilize stochastic gradient updating strategies to reduce per-iteration computation cost. For example, RDA-ADMM \cite{126} incorporates RDA method with ADMM; SADMM and optimal-SADMM in \cite{3} utilize nonuniform averaging of iterative variable \cite{78,121} and accelerated stochastic gradient method \cite{13} to further accelerate the stochastic ADMM method. Incremental ADMM methods \cite{158,127} can achieve faster convergence rate than that of stochastic ADMM by utilizing the incremental gradient updating strategy. In particular, SA-ADMM \cite{158} and SDCA-ADMM \cite{127} are two recently proposed ADMM methods incorporating two different incremental gradient methods: SAG \cite{116} and SDCA \cite{120} correspondingly. However, despite the above effort for better efficiency and scalability, a remaining major drawback of ADMM-based methods is the complex implementation and convergence analysis, which are brought about by the additional variables introduced and the alternating updating scheme.

Recently, an alternative to ADMM called proximal average (PA) \cite{146} has been introduced to efficiently handle composite penalties. It approximates the original composite penalty when each constituent regularizer admits simple proximal map. The resulting proximal average approximation then enjoys simple proximal map by averaging the proximal map of its components. What make the PA technique be interesting are that the approximation can be controlled arbitrarily close to the original composite regularizer and be strictly better than smoothing technique. Compared with ADMM, \cite{157} points out that ADMM is also a proximal method by duplicating variables. Apparently, proximal average is much simpler to implement.
and also much easier to make analysis, which will be introduced later. Along this
time, pioneer work includes the one in [146], which introduces proximal average with
accelerated full gradient method FISTA [7]. [157] incorporates proximal average
technique with the stochastic variant of optimal gradient method. It has provable
superiority over smoothing technique which is also shared by [146]. Despite the
simplicity advantage in terms of implementation and analysis, when compared
to incremental ADMM methods (e.g. SA-ADMM and SDCA-ADMM), existing
PA-based approaches either converge slowly (e.g. PA-ASGD) or suffer from high
per-iteration cost (e.g. PA-APG).

Incremental gradient methods featuring both scalability and fast convergence
property have been receiving considerable attention as an efficient approach to miti-
gating the ever growing dataset problem. As these methods only calculate gradients
associated with a randomly picked data sample in each iteration as stochastic gradient
methods [11, 139, 43], they have comparable low per-iteration computation cost.
More importantly, by well exploiting the finite sum structure of the loss function
which stochastic methods do not have, these incremental methods are able to achieve
linear convergence rate as full gradient methods [103]. For example, SAG [116]
utilizes the average of the stored past gradients, one for each data. SVRG [66, 140]
adopts a multi-stage scheme to progressively control the variance of the stochastic
gradient. Both methods have linear convergence rate for strongly convex problem,
but the theoretical convergence result for general convex loss is still unclear by now.
SAGA ([25, 26]) has both sublinear convergence guarantee for general convex loss
and linear convergence for strongly convex loss. It is a midpoint of SAG and SVRG
by taking both update pattern from them in its iteration. There are also other
incremental methods like FINITO [27] and MISO [91], which consume more memory
because they not only store the gradient, but also the variable. S2GD [76] is a
method very similar to SVRG with the difference only in stage length. SDCA [120]
is a dual incremental method.

The above-mentioned methods mainly focus on convex composite penalties.
Nonconvex composite penalties, although leading to an even more difficult problem, can have better prediction performance by avoiding the over-penalization problems of their convex correspondences. For structured sparsity inducing tasks, there have been some research incorporating structured sparsity regularizers with nonconvex and showing their improved prediction performance \cite{122, 138}. For optimizing such nonconvex composite penalties, general nonconvex solvers like concave-convex procedure (CCCP) \cite{151} and sequential convex program (SCP) \cite{89} proceed in a multi-stage convexify scheme that solves a convex relaxation in each stage up to certain approximation and then constructs a convex surrogate for the next stage. \cite{159} has recently proposed a proximal average based gradient descent method called GD-PAN for such penalty. It has been shown that it is still possible to approximate the nonconvex composite function with proximal average for some common nonconvex penalties. Also, by solving such surrogate, it is more efficient than multi-stage methods like CCCP and SCP, because the proximal map of the proximal gradient descent can be easily computed for the surrogate. However, GD-PAN that is essentially a batch gradient method suffers from the scalability problem.

In this chapter, we also propose an incremental proximal average method for solving nonconvex composite penalty problems.

In this chapter, we shall investigate the potential to incorporate incremental gradient methods with proximal average technique. For the convex composite penalties, we will show that, by solving a surrogate problem, the proposed method can achieve linear convergence when the loss function is strongly convex and sublinear convergence rate when the loss is general convex. By contrast, ADMM-based methods cannot provide both. For example, SDCA-ADMM only has convergence results for strongly convex loss, while the convergence analysis of SAG-ADMM only applies when the loss is general convex. Furthermore, we also extend the incremental PA technique to solve nonconvex penalty problems, which have better scalability than that of batch method GD-PAN \cite{159}. In this setting, we will show that the proposed method converges to asymptotic stationary point of the surrogate problem.
The remainder of this chapter is organized as follows: Section \ref{sec:preliminaries} introduces notations and assumptions used in this chapter. Section \ref{sec:pa} conducts an overview of PA and incremental gradient descent methods. In Section \ref{sec:convex}, we propose our method for convex composite penalties with strongly convex loss and general convex loss, and establish the corresponding convergence rate. Section \ref{sec:nonconvex} proposes an incremental proximal average algorithm for solving nonconvex composite penalty problems. Section \ref{sec:results} shows the experimental results for both convex composite penalty problems and nonconvex composite penalty problems on synthetic and real datasets. Finally, Section \ref{sec:conclusion} summarizes the chapter.

\section{Preliminaries}

In this section, we firstly introduce the notations used in this chapter. Then, we formally define the problem to be optimized. Also, we will describe the assumptions for these problems.

\textbf{Additional Notations} In the following, we denote by $\langle \nabla l_i(x), y \rangle$ the inner product of $\nabla l_i(x)$ and $y$. The superscript $(\cdot)^T$ stands for the transpose of $(\cdot)$. We denote the $t$-th iteration of $x$ by $x^t$. We assume the dataset is indexed as $1, 2, ..., N$, and the subscript $i$ like $x_i$ is related to the $i$-th data sample. We denote the $k$-th component of the composite penalty function by the subscript $k$ in $r_k$.

We consider the following ERM with composite penalty problem:

$$
\min_{x \in \mathbb{R}^d} F(x) = l(x) + r(x) = \frac{1}{n} \sum_{i=1}^{n} l_i(x) + \sum_{k=1}^{K} \alpha_k r_k(x),
$$

(4.2.1)

$$
\sum_{k=1}^{K} \alpha_k = 1, \quad \alpha_k \geq 0,
$$

(4.2.2)

which is commonly applied to learn the model defined by variable $x$ from training data set $\{\xi_i, y_i\}$ $i = 1, ..., n$. $\xi_i$ is the data vector, and $y_i$ is its label. In eq. (4.2.1), $l_i(x)$ is the loss taken at data sample $(\xi_i, y_i)$ with index $i$. The function $r(x)$ is the composite penalty for regularization purpose, which is composed by $K$ constituent regularizers. We hide the constant balancing the loss and the regularizer in the loss.
as [146] and [157], so that $r(x)$ is a convex combination of the $K$ components $r_k(x)$. In this chapter, we allow both $l(x)$ and $r(x)$ to be either convex or nonconvex.

**Smooth loss function:** In this chapter, we assume $l_i(x)$ to be smooth with $L$ Lipschitz continuous gradient, so that we can take the gradient for gradient descent and also we are able to construct a local majorization surrogate. Formally, an $L$-smooth loss function $l_i$ satisfies the following inequality,

**Assumption 4.2.1.** The loss function is $L$-smooth, $\forall x, y,$

$$l_i(y) - l_i(x) - \langle \nabla l_i(x), y - x \rangle \leq \frac{L}{2} ||y - x||^2. \quad (4.2.3)$$

If we further assume $l_i(x)$ is general convex, $l_i$ also satisfies the following inequality:

**Assumption 4.2.2.** $l_i(x)$ is convex if $\forall x, y,$

$$l_i(y) - l_i(x) - \langle \nabla l_i(x), y - x \rangle \geq 0. \quad (4.2.4)$$

Examples of the general convex smooth loss functions include least square loss, logistic loss, and smooth hinge loss, all of which will be used in Section 4.5. In addition, $l_i(x)$ can be strongly convex provided that the following assumption is held:

**Assumption 4.2.3.** $l_i(x)$ is strongly convex if there is a $\mu > 0$ such that $\forall x, y,$

$$l_i(y) - l_i(x) - \langle \nabla l_i(x), y - x \rangle \geq \frac{\mu}{2} ||y - x||^2. \quad (4.2.5)$$

For example, when combining the above general convex loss with a large margin inducing penalty $\frac{1}{2} ||x||^2$, it becomes a $\lambda$-strongly convex loss.

**Composite Penalty:** We focus on composite penalty in this chapter, i.e. $r(x)$ is an average of $K$ simple non-smooth penalties $r_k(x)$. We assume that $r_k$ is Lipschitz continuous with the constant $M_k$, i.e.

**Assumption 4.2.4.** $r_k$ is $M_k$ Lipschitz continuous, $\forall x, y,$

$$|r_k(x) - r_k(y)| \leq M_r||x - y||_2. \quad (4.2.6)$$
Also, the proximal update step of each $r_k$ should be simple. Please note that the proximal map of $r(x)$ itself can be very complex and computationally expensive. In addition, we introduce the notations related to proximal step:

\[
M^\eta_{r_k}(x) = \min_y \frac{1}{2\eta} \|x - y\|^2_2 + r_k(y),
\]

and

\[
P^\eta_{r_k}(x) = \arg \min_y \frac{1}{2\eta} \|x - y\|^2_2 + r_k(y).
\]

### 4.3 Accelerated Proximal Average Approximated Incremental Gradient for ERM with Convex Composite Penalty

In this section, we present the proposed incremental gradient descent proximal average method for convex composite penalty regularized ERM problems, which is termed as IncrePA-cvx. We first illustrate the convex composite penalty functions with two types of structured sparsity inducing penalties as examples, i.e. overlap group lasso and graph-guided lasso. We then describe the proposed method provided with the convergence rate for convex composite penalties with general convex and strongly convex loss.

#### 4.3.1 Overlapping Group Lasso and Graph-guided Fused Lasso

In the following, we describe two convex composite regularizers for inducing structured sparsity among features in sparsity estimation tasks.

**Overlapping group lasso:** [58] introduces overlapping group lasso

\[
r(x) = \sum_{k=1}^{K} \omega_k r_k(x) = \sum_{k=1}^{K} \frac{1}{K} \|x_{g_k}\|_2,
\]
where \( g_k \) indicates the index group of features, and \( x_{g_k} \) is a copy of \( x \) with the values of those that are not in the index subset \( g_k \) being set at 0. Apparently, the proximal map of each individual \( ||x_{g_k}|| \) is simple to compute, while the proximal map of \( r(x) \) is difficult due to the coupled nature of overlapping groups of indices.

**Graph-guided fused lasso:** \( \zeta \) induces structured sparsity according to the graph \( \mathcal{E} \),

\[
    r(x) = \sum_{k=1}^{K} \omega_k r_k(x) = \sum_{k=1}^{K} \omega_k |x_{k_1} - x_{k_2}|, \tag{4.3.10}
\]
where \( \{k_1, k_2\} \in \mathcal{E} \). Again, the proximal map of \( r(x) \) is not easy to compute even though \( r_k(x) \) is.

### 4.3.2 Incremental Gradient Proximal Average for Convex Composite Penalty Regularized ERM

The proposed method proceeds with proximal gradient style iterative scheme. With the estimated gradient utilized in iteration \( t \) denoted by \( G^t \) and step size by \( \eta \), the algorithm updates:

\[
x_{t+1} = \arg\min_x \frac{1}{2\eta} ||x - (x_t - \eta G^t)||_2^2 + r(x), \tag{4.3.11}
\]
which can be denoted by \( x^{t+1} = P^\eta_r(x^t - \eta G^t) \) (recall that the proximal map corresponding to penalty function \( r(x) \) is denoted by \( P^\eta_r(\cdot) \), as shown in eq. (4.2.8)). The gradient \( G^t \) is estimated by incremental gradient strategy, in particular SAGA \( \zeta \), which consumes low per-iteration cost and fast convergence by reducing the variance of the estimated gradient.

In general, the proximal map corresponding to composite penalties \( r(x) \) is not easy to compute. Popular approaches propose to deal with composite penalty functions based in splitting method ADMM. When coupled with incremental gradient estimation strategies, ADMM-based methods are difficult to be analyzed. For example, SA-ADMM (based on SAG and linearized ADMM) only has convergence results when the loss function is general convex, while SDCA-ADMM only has
convergence results when the loss function is locally strongly convex. Most recently, SVRG-ADMM \[155\] is able to provide the convergence analysis for both general and strongly convex losses, but they require different iteration design under different convexity assumption. Hence, to develop a general algorithm that is capable to cover both general and strongly loss function case with unified iteration scheme, we propose to approximate \( r(x) \) with proximal average approximation \( \hat{r}(x) \). The iteration becomes

\[
x^{t+1} = P^\eta_\hat{r}(x^t - \eta G^t),
\]  
(4.3.12)

which can be simply computed according to the proximal average property as shown in eq. \[(2.2.14)\] by

\[
x^{t+1} = \sum_{k=1}^{K} \alpha_k P^\eta_{r_k}(x^t - \eta G^t),
\]  
(4.3.13)

where \( P^\eta_{r_k}(\cdot) \) is the proximal mapping with respect to simple constituent function \( r_k \). By utilizing the proximal average update technique, we are actually solving the surrogate problem \( \hat{f}(x) = l(x) + \hat{r}(x) \), which can be controlled arbitrarily close to the original problem \( F(x) \) according to lemma \[(2.2.2)\]. We summarize the closeness property by the following lemma.

**Lemma 4.3.1.** For \( r(x) \) with \( M_k \)-Lipschitz continuous \( r_k(x) (k = 1, ..., K) \) and denote \( \bar{M}^2 \)

\[
= \sum_{k=1}^{K} \alpha_k M_k^2
\]

as in lemma \[(2.2.2)\], we have \( F(x) - \hat{f}(x) \leq \epsilon \) for any \( x \), when we set \( \eta \leq \frac{2\epsilon}{\bar{M}^2} \).

The proposed incremental gradient PA for ERM with convex composite penalty is summarized in Algorithm \[10\].
Algorithm 10 IncrePA-cvx

Input: $\eta$ (step size); $x_0$ (initial value); $\nabla l_i(\phi^0_i), \phi^0_i = x_0, i = 1, ..., n$ (initial table of gradients).

1: for $t = 0, 1, 2, ...$ do
2: Randomly pick $i^t \in \{1, 2, ..., n\}$;
3: Update the derivative table as in equation (2.2.16);
4: Calculate $G^t$ by equation (2.2.17);
5: $w^{t+1} = x^t - \eta G^t$;
6: $x^{t+1} = \sum_{k=1}^{K} \alpha_k P_{r_k}^\eta(w^{t+1})$;
7: end for

4.3.3 Analysis of IncrePA-cvx

The proposed method is general in the sense that it is provided with convergence analysis covering both general convex loss and strongly convex loss functions cases with the unified iteration design. We describe them as follows.

A: Convergence Analysis for General Convex Objectives: In this paragraph, we establish the convergence rate of IncrePA when applied to general convex objectives. Recall the notation of the surrogate function $\hat{f}(x) = l(x) + \hat{r}(x)$ implicitly solved by IncrePA. The following theorem summarizes the sublinear convergence rate:

Theorem 4.3.2. Under Assumption 4.2.1 (i.e. $l_i$ is smooth) with $l_i(x)$ general convex and Assumption 4.2.4 (i.e. $r_k$ is simple and Lipschitz continuous), let $\hat{x}^*$ be the optimal point of the surrogate problem. Denote $Q^t$ as

$$Q^t = \frac{1}{N} \sum_{i=1}^{n} l_i(\phi_i^t) - l(\hat{x}^*) - \frac{1}{N} \sum_{i=1}^{n} \langle \nabla l_i(\hat{x}^*), \phi_i^t - \hat{x}^* \rangle.$$

Then, after $t \geq \frac{1}{\epsilon^2} \left( Q^0 + (c_1 + \frac{c_2}{2\eta}) \|x^0 - \hat{x}^*\|_2^2 \right)$ iterations, we have

$$\mathbb{E} \left[ F(x^t) - F(\hat{x}^*) \right] \leq 2\epsilon,$$

54
where $\bar{x}^t = \frac{1}{t} \sum_{i=1}^{t} x^i$. In addition, possible choices of the parameters $c_1, c_2, \eta$ appeared in the proof are as follows: $\eta < \min(\frac{1}{2L}, \frac{2\kappa}{M^2})$, $c_1 = \frac{1}{2nN}$, $c_2 = \frac{1}{2N} \left(\frac{1}{2\eta L\beta} - 1\right)$.

**B. Convergence Analysis for Strongly Convex Objectives:** If we further have the strongly convexity of the loss function, the proposed method can achieve linear convergence as shown in Theorem 4.3.3.

**Theorem 4.3.3.** Under Assumption 4.2.1 (i.e. $l_i$ is smooth) with $l_i(x)$ $\mu$-strongly convex and Assumption 4.2.4 (i.e. $r_k$ is simple and Lipschitz continuous), let $\hat{x}^*$ be the optimal point of the surrogate problem. Denote a Lyapunov function $T^t$ as:

\[
T^t = Q^t + (c_1 + \frac{c_2}{\eta}) \|x^t - \hat{x}^*\|^2_2 + c_2 (\hat{f}(x^t) - \hat{f}(\hat{x}^*)),
\]

(4.3.16)

\[
Q^t = \frac{1}{n} \sum_{i=1}^{n} l_i(\phi_i^t) - l(\hat{x}^*) - \frac{1}{N} \sum_{i=1}^{n} \langle \nabla l_i(\hat{x}^*), \phi_i^t - \hat{x}^* \rangle,
\]

(4.3.17)

where $t$ is the iteration number. After $(1 - \frac{1}{\kappa}) (\log \frac{T^0}{\epsilon})$ iterations, we then have

\[
\mathbb{E} \left[ F(x^t) - F(\hat{x}^*) \right] \leq 2\epsilon.
\]

(4.3.18)

In addition, there exits some $\beta \geq 1$ and possible choices of the parameters $c_1, c_2, \kappa, \eta$ appeared in the proof are as follows: $\eta < \min(\frac{1}{2L}, \frac{2\kappa}{M^2}, \frac{1}{2N\mu})$, $c_1 = \frac{1}{2nN} \frac{L}{L-\mu}$, $c_2 = c_1 \eta \left(\frac{1}{2\eta L\beta} - 1\right)$, $\frac{1}{\kappa} = \frac{2\eta\mu}{1 + \frac{2n\eta}{L\mu}}$.

The detailed mathematical proof of Theorem 4.3.2 and Theorem 4.3.3 is given in Appendix.

**4.3.4 Discussion**

We have the following three remarks to discuss about the above two convergence guarantees.

**Remarks 1:** First we point out the specialties of the step size parameter $\eta$. One can find that we represent all parameters by the step size $\eta$ in the above convergence analysis because it controls the approximation by Lemma 2.2.2. The convergence rate of strongly convex case is related to $\frac{1}{\kappa}$, i.e. it converges faster when $\frac{1}{\kappa}$ is larger, which
depends on $L, \mu$ and $\epsilon$ as given the data size $n$. Please note that, for an ill-conditioned problem where $\frac{L}{\mu} = N$, $\frac{1}{2N\mu}$ can be converted to $\frac{1}{2L}$. Thus, the convergence speed is related to $L$ and $\epsilon$. The convergence speed for the general convex case depends on $c_2$, i.e. the larger $c_2$ is, the faster it converges. Given the dataset size, the convergence speed is again related to $L$ and $\epsilon$.

**Remarks 2:** Like the other incremental methods, the above convergence only reflects training loss ([127], [158], [117]). The generalization performance is unknown partly because of the assumption of the finite training set size. Our experiments on testing loss show empirical results of the generalization performance.

**Remarks 3:** Furthermore, our algorithm will converge to the optimal point of the surrogate function. We show the convergence rate by measuring the loss with respect to the objective function value at $\hat{x}^* (F(\hat{x}^*))$, which is different from usual convention that measures with $F(x^*)$. Nevertheless, considering the over-fitting issue, a relative good approximation is potentially able to achieve satisfactory generalization performance. As a good approximation to the original problem, it is expected that the proposed method will have satisfactory generalization performance. Indeed, the experimental results in Section 4.5 have verified this in terms of classification error and test loss on the test set of two real datasets.

### 4.4 Incremental Proximal Average for Nonconvex Composite Penalty Regularized ERM

In this section, we extend the incremental gradient with proximal average algorithm to nonconvex composite penalty regularized ERM problems. We first describe example nonconvex composite penalties utilized in structured sparsity estimation tasks, which replace the convex $\ell_1$ norm with tighter nonconvex surrogate functions of the $\ell_0$ norm. After recalling existing approaches for this type of problems, we present a more scalable method by extending the IncrePA-cvx in the previous section to nonconvex...
composite penalty case, termed as IncprePA-ncvx.

### 4.4.1 Two Examples of Nonconvex Composite Penalties in Structured Sparse Estimation

Nonconvex composite penalties appear in nonconvex structured sparsity estimation applications. The nonconvex surrogate penalties like capped $\ell_1$ norm, smoothly clipped absolute deviation (SCAD) and minimax concave penalty (MCP), are able to address the biasness of the convex $\ell_1$ norm, thus are considered better relaxations of the $\ell_0$ norm for promoting sparsity. Inspired by this, papers [122, 138, 159] have proposed nonconvex structured sparsity inducing counterparts by wrapping the convex composite functions with the nonconvex functions. That is, the penalty function $r(x)$ takes the composite form as an average of $K$ nonconvex composite penalties as

$$ r(x) = \sum_{k=1}^{K} \omega_k r_k(x). \quad (4.4.19) $$

In this nonconvex composite penalty case, each $r_k$ takes the following form:

$$ r_k(x) = \rho(h_k(x)), \quad (4.4.20) $$

where $\rho(\cdot)$ is the nonconvex sparsity-inducing function.

In this structured case, compared with traditional non-structured nonconvex relaxations of lasso, it is wrapped outside each constituent convex regularizers rather than each indices of $x$. We elaborate eq. (4.4.19) with Capped-$\ell_1$ overlapping group-lasso and MCP graph-guided fused Lasso as two concrete examples.

**Capped-$\ell_1$ overlapping group-lasso:** This is a hybrid nonconvex composite penalty of Capped-$\ell_1$ norm and overlapped group-lasso, which wraps each group indices $h_k(x) = ||x_{g_k}||_2$ with Capped-$\ell_1$ norm [131]:

$$ r(x) = \sum_{k=1}^{K} \omega_k r_k(x) = \sum_{k=1}^{K} \omega_k \rho(||x_{g_k}||_2) = \sum_{k=1}^{K} \omega_k \min\{||x_{g_k}||_2, \theta\}, \quad (4.4.21) $$

where $\theta$ is a constant defining the $\ell_1$ norm.
MCP graph-guided fused Lasso: This nonconvex composite penalty combines MCP with graph-guided fused lasso.

\[ r(x) = \sum_{k=1}^{K} \omega_k r_k(x) = \sum_{k=1}^{K} \omega_k \rho(|x_{k_1} - x_{k_2}|), \]  

(4.4.22)

where \( \{k_1, k_2\} \in \mathcal{E}, |\mathcal{E}| = K \) and \( \rho(\cdot) \) takes the following form based on MCP norm:

\[
\rho(u) = \begin{cases} 
\lambda |u| - \frac{u^2}{2a}, & |u| \leq a\lambda, \\
\frac{a\lambda^2}{2}, & |u| > a\lambda,
\end{cases}
\]

(4.4.23)

where \( \lambda \) and \( a \) are constants.

4.4.2 Related Work

Such composite form and nonconvexity make the problem even more difficult to solve. Some existing approaches are proposed with inefficiency or scalability issues. DC programming-based methods like concave-convex procedure (CCCP) progress by stages that solve a convex surrogate in each stage by approximating nonconvex \( r(x) \) with a convex function. This multistage style can be inefficient. General iterative shrinkage and thresholding (GIST) and sequential convex program (SCP) can be efficient for regularizers with simple proximal update. However, since the proximal step is very difficult for eq. (4.4.19), these methods are also not efficient enough. Recently, GD-PAN has extended proximal average for nonconvex eq. (4.4.19) and approximates \( r \) with proximal average in the GIST algorithm to obtain a proximal update efficient algorithm. However, GD-PAN is intrinsically a batch gradient algorithm with poor scalability towards large-scale problems. Apparently, a PA-based method with better scalability is more attractive and useful from a practical perspective.

4.4.3 Nonconvex Extension of Incremental Gradient with PA

We aim to extend the incremental gradient PA method to solve these nonconvex structured problems, termed as IncrePA-ncvx. Also approximates the nonconvex
composite regularizer with PA, and then solves the approximate problem based on iteration scheme, which is a batch gradient method. Our method improves upon with incremental gradient strategy that results into better scalability.

In this nonconvex case, we also approximate $r(x)$ with its PA approximation $\hat{r}(x)$, which is similar to convex case. For convenience, we denote the PA approximated objective as

$$\arg\min_{x} \hat{f}(x) = \arg\min_{x} \frac{1}{N} \sum_{i=1}^{N} \hat{f}_i(x) = \arg\min_{x} \frac{1}{N} \sum_{i=1}^{N} [l_i(x) + \hat{r}(x)], \quad (4.4.24)$$

where each component function $\hat{f}_i(x) = l_i(x) + \hat{r}(x)$ corresponds to the $i$-th data sample. The PA approximated function $\hat{f}(x)$ is not guaranteed to be convex. Hence, directly applying incremental proximal gradient decent method to $\hat{f}_i(x)$ can hardly ensure convergence. In this regard, we further approximate $\hat{f}(x)$ iteratively with the first-order surrogate of $\hat{f}(x)$ by following \[91\], which is a particular majorization by taking the smoothness of $l_i(x)$ into consideration. Again, as an incremental method, we keep a variable table and a gradient table, in which we denote them again by $\phi_i^t$ and $\nabla l_i(\phi_i^t)$ correspondingly for the $i$-th sample at iteration $t$, by the random choose-and-replace strategy as in the previous section. At iteration $t$, with the latest variable table and gradient table, a majorization approximation $g_i^t(x)$ of $\hat{f}_i(x)$ is constructed as

$$g_i^t(x) = l_i(\phi_i^t) + \langle \nabla l_i(\phi_i^t), x - \phi_i^t \rangle + \frac{1}{2\eta} \|x - \phi_i^t\|_2^2 + \hat{r}(x), \quad (4.4.25)$$

where $\eta$ is the step size and satisfies $\frac{1}{\eta} \geq L$. By the smoothness assumption of the loss function $l_i(x)$ (assumption \[4.2.3\]), function $g_i^t(x)$ upper bounds $\hat{f}_i(x)$ (i.e. $g_i^t(x) \geq \hat{f}_i(x)$). Then, in each iteration, the majorization function is minimized with

$$x^{t+1} = \arg\min_{x} g^t(x) = \arg\min_{x} \frac{1}{N} \sum_{i=1}^{N} g_i^t(x)$$

$$= \arg\min_{x} \frac{1}{N} \sum_{i=1}^{N} \left[ l_i(\phi_i^t) + \langle \nabla l_i(\phi_i^t), x - \phi_i^t \rangle + \frac{1}{2\eta} \|x - \phi_i^t\|_2^2 + \hat{r}(x) \right], \quad (4.4.26)$$

which is an incremental majorization-minimization iteration by choosing the majorization function as the so-called first-order surrogate \[91\]. With such surrogates during
iteration, we need to extra memory to explicitly store the variable table as compared with convex incremental gradient PA method, where the variable table is introduced only for notational convenience and need not be kept. However, this overhead in memory seems indispensable, because the per-iteration problem evaluated in the previous section cannot be guaranteed to be majorization of \( \hat{f}(x) \), which is obvious when we rewrite the iterate scheme of Algorithm 10 in the same style as eq. (4.4.26),

\[
x^{t+1} = \arg \min_x \frac{1}{N} \sum_{i=1}^{N} \left[ l_i(x_t) + \langle \nabla l_i(x_t), x - x^t \rangle + \frac{1}{2\eta} \|x - x^t\|_2^2 + \hat{r}(x) \right]. \tag{4.4.27}
\]

Then, eq. (4.4.26) can be further simplified to \( x^{t+1} = \arg \min_x \frac{1}{N} \sum_{i=1}^{N} \phi_i^t - \eta G^t \| x - \frac{1}{N} \sum_{i=1}^{N} \phi_i^t \|_2^2 + \hat{r}(x) \), where \( G^t = \frac{1}{N} \sum \nabla l_i(\phi_i^t) \). By the property of the PA approximation function \( \hat{r}(x) \) and the proximal mapping notation as in eq. (2.2.14), we then have

\[
x^{t+1} = \sum_{k=1}^{K} \alpha_k P_{r_k}^\eta \left( \frac{1}{N} \sum_{i=1}^{N} \phi_i^t - \eta G^t \right). \tag{4.4.28}
\]

We summarize the above iteration scheme by IncrePA-ncvx as shown in Algorithm 11.

**Algorithm 11 IncrePA-ncvx**

**Input:** \( \eta \) (step size); \( x^0 \) (initial variable); \( \nabla l_i(\phi_i^0), i = 1, ..., n \) (initial table of gradients); \( \phi_i^0, i = 1, ..., n \) (initial table of iterate x).

1. for \( t = 0, 1, 2, ... \) do
2. Randomly pick \( i^t \in \{1, 2, ..., n\} \);
3. Update the derivative table as in eq. (2.2.16);
4. Update the variable table as in eq. (2.2.15);
5. Calculate \( G^t \) by averaging the gradient table;
6. \( w^{t+1} = \frac{1}{N} \sum_{i=1}^{N} \phi_i^t - \eta G^t; \)
7. \( x^{t+1} = \sum_{k=1}^{K} \alpha_k P_{r_k}^\eta (w^{t+1}); \)
8. end for

4.4.4 Analysis of IncrePA-ncvx

The main per-iteration computational cost comes from: i) step 5 evaluates a stochastic gradient and ii) step 7 computes the proximal mapping with respect to \( K \) simple
regularizers and takes the average. Hence, compared to the PA-based method GD-PAN, the proposed method provides better scalability when the dataset size grows because the per-iteration computational cost is irrelevant to the number of data points.

For nonconvex problems, it is generally impossible to guarantee global optimum or derive convergence rate as those for convex and strongly convex problems. Following [91], we only provide the convergence of IncrePA-ncvx in the sense that the PA approximation \( \hat{f}(x^t) \) is almost sure convergence and the sequence \([x^t]\) satisfies the so-called asymptotic stationary point condition (for more details, see [10]).

**Definition 4.4.1.** Asymptotic stationary point: Denote the directional derivative of function \( f \) at \( x^t \) as \( \nabla f(x^t, x - x^t) \) (see subsection 2.1 in [10] for detailed definition), under the assumption that \( f \) is bounded below and for all \( x, x^t \), the directional derivative \( \nabla f(x, x - x^t) \) of \( f \) at \( x^t \) in the direction \( x - x^t \) exists, the sequence \([x^t]_{t=1,2,...}\) satisfies the asymptotic stationary point condition if

\[
\lim_{k \to \infty} \inf_{x \in X} \inf_{x^t} \frac{\nabla f(x^t, x - x^t)}{||x - x^t||_2} \geq 0. \tag{4.4.29}
\]

We rely on the convergence result from [91], through which we have the following lemma:

**Lemma 4.4.2.** Suppose \( f(x) = \sum_{i=1}^{N} f_i(x) \) is bounded below and the directional derivative exits. With \( g_{\epsilon}^t(x) \) being first-order surrogates and incremental majorization-minimization scheme, \( f(x^t) \) is almost sure convergence and \( x^t \) satisfies the asymptotic stationary point condition with probability one.

Based on Lemma (4.4.2), we have the convergence result for our IncrePA-ncvx as summarized in the following Theorem (4.4.3)).

**Theorem 4.4.3.** Algorithm IncrePA-ncvx is almost sure convergence and the iterates \( x^t \) of converges to the asymptotic stationary point of the surrogate problem \( \hat{f}(x) \) with probability one.

**Proof.** To utilize Lemma 4.4.2, we first observe that our surrogate function \( g_{\epsilon}^t(x) \) in eq. (4.4.25) is the so-called first-order surrogate of the PA approximation function
\( \hat{f}_i(x) \) in eq. (4.4.24). Namely, i) \( g_i^t(x) \) majorizes \( f_i^t(x) \), i.e. \( g_i^t(x) \geq \hat{f}_i(x) \); ii) Denote the approximation error by \( h_i^t(x) = g_i^t(x) - \hat{f}_i^t(x) = l_i^t(\phi_i^t) - l(x) + \langle \nabla l_i^t(\phi_i^t), x - \phi_i^t \rangle + \frac{1}{2}\|x - \phi_i^t\|_2^2 \), then \( h_i^t(x) \) is smooth and \( h_i^t(\phi_i^t) = 0, \nabla h_i^t(\phi_i^t) = 0 \). Hence, with the first-order surrogate adopted in IncrePA-ncvx and the incremental majorization-minimization scheme, we can apply Lemma 4.4.2 for the sequence \( \hat{f}(x^t) \) and \( x^t \) to conclude that IncrePA-ncvx is almost sure convergent to the asymptotic stationary point of proximal approximation function \( \hat{f}(x) \) with probability one.

4.5 Experiments

In this section, we evaluated the empirical performance of IncrePA for both convex composite penalty and nonconvex composite penalty. We implemented the proposed method and all other methods for comparison in MATLAB. All experiments were conducted on single core of a laptop and 2.6-GHz Intel CPU with 16 GB of RAM. We used both synthetic datasets and four real datasets\(^1\) in the experiment on the task basis. The real datasets are summarized in Table 4.1. We randomly sampled 80% of the data as training set and the rest as testing set. We used four different tasks to demonstrate the performance of the proposed method according to the convexity or nonconvexity of the composite penalty and general or strongly convexity of the loss function. As a result, we provided empirical evidence for all kinds of combinations of loss functions and penalties, to which the proposed IncrePA has provided theoretical convergence results in the previous sections. In the following, we have:

- Subsection 6.1 considers general convex loss with convex composite penalty by solving smooth hinge loss with graph-guided lasso task on four real datasets;

- Subsection 6.2 considers strongly convex loss with convex composite penalty by solving logistic loss with large margin graph-guided lasso on four real datasets;

- Subsection 6.3 considers nonconvex composite penalty of capped \( \ell_1 \) norm

\(^1\)‘a9a’ and ‘covtype’ are from LIBSVM archive; ‘protein’ is from KDD CUP 2004; ‘20 newsgroup’ is from [http://www.cs.nyu.edu/~roweis/data.html](http://www.cs.nyu.edu/~roweis/data.html)
overlapping group lasso on synthetic datasets with the different number of
groups and data points.

• Subsection 6.4 considers nonconvex composite penalty of capped \(\ell_1\) norm
graph-guided lasso on four real datasets.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Data points</th>
<th>Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 newsgroup</td>
<td>12,995</td>
<td>100</td>
</tr>
<tr>
<td>a9a</td>
<td>32,561</td>
<td>123</td>
</tr>
<tr>
<td>covtype</td>
<td>581,012</td>
<td>54</td>
</tr>
<tr>
<td>protein</td>
<td>145,751</td>
<td>74</td>
</tr>
</tbody>
</table>

### Experiment 1: Solving general convex loss function with convex composite penalty

In this and the next subsections, we evaluated the performance of IncrePA on
convex composite penalties in comparison with two incremental gradient ADMM:
SA-ADMM \[158\] and SDCA-ADMM \[127\] along with a PA-based stochastic gradient
PA-ASGD \[157\]. We do not consider the batch gradient PA method for comparison
because \[157\] has already shown that it is less efficient than PA-ASGD. Also, we do
not explicitly compare the proposed algorithm with the stochastic ADMM methods
because the latter is slower than the incremental ADMM methods as demonstrated
in \[158\] and \[127\].

In this subsection, we considered the general convex loss problem by using the
smoothed hinge loss:

\[
l_i(u) = \begin{cases} 
0, & y_i u \geq 1 \\
\frac{1}{2} - y_i u, & y_i u \leq 0 \\
\frac{1}{2}(1 - y_i u)^2, & \text{otherwise},
\end{cases}
\] (4.5.30)
Figure 4.1: General convex loss with convex composite penalty: Empirical risk on training data versus effective passes of smooth hinge loss with graph-guided lasso on four real datasets.

where \( u = \xi_i^T x \), \((\xi_i, y_i)\) is the i-th data sample. We utilized the graph-guided fused lasso

\[
\lambda (||x||_1 + \sum_{\{i,j\} \in E} |x_i - x_j|)
\]

as the convex composite regularizer. We constructed the graph by sparse inverse covariance matrix as used in [127] and set \( \lambda \) at 0.001. The proximal map for \( ||x||_1 \) is simply soft thresholding. The proximal map for \( |x_i - x_j| \) is

\[
[P^\eta_{rk}]_s = \begin{cases} 
  x_s - \text{sign}(x_i - x_j) \min\{\eta, \frac{|x_i - x_j|}{2}\}, & s \in \{i, j\} \\
  x_s, & \text{otherwise}
\end{cases}
\]

as given in [146 157]. For the training performance, we reported the empirical risk, which is the training loss, against the number of iterations for all datasets in Figure 4.1. As for the generalization performance, we reported the classification
Figure 4.2: General convex loss with convex composite penalty: Classification error on testing data versus effective passes and CPU time of smooth hinge loss with graph-guided lasso on two real datasets.

As shown in Figure 4.1, in terms of reducing the empirical loss, the performance of the proposed method is the best on ‘20 newsgroup’ and ‘protein’, and only falls to SDCA-ADMM on ‘covtype’ and is only inferior to SA-ADMM on ‘a9a’. On all datasets, IncrePA is more efficient than another PA-based method: PA-ASGD. Therefore, in this task, IncrePA performs almost the same as the other two ADMM-based incremental gradient methods and is a much faster PA-based method compared with the PA-ASGD. Figure 2 demonstrates the generalization performance. When compared against the iteration numbers, IncrePA performs similar with SA-ADMM, which is better than PA-ASGD, although both are a bit inferior to SDCA-ADMM. When compared against CPU time, the proposed method performs relatively better.
than the other methods on both datasets. As a conclusion, IncrePA has the similar generalization performance in terms of classification error on both dataset with SA-ADMM and SDCA-ADMM and is more efficient than PA-ASGD. Also, the classification error on both testing sets indicate that our solution obtained by the surrogate to regularizer is able to achieve satisfactory generalization performance.

4.5.2 Experiment 2: Solving strongly convex loss function with convex composite penalty

For strongly convex case, we utilized the logistic loss with the large margin graph-guided lasso regularizer as in (157), i.e.

$$\lambda(||x||^2_2 + \sum_{\{i, j\} \in E} |x_i - x_j|).$$

(4.5.33)
According to Figure 3, our method performs relatively better on ‘20 Newsgroup’ and ‘a9a’, and is similar with SA-ADMM on ‘covtype’ and ‘protein’ in training. As for the generalization performance, Figure 4 shows the decrease of test loss over iteration number and CPU time. IncrePA performs better than the other two methods on ‘a9a’ in terms of both number of iterations and CPU time. IncrePA is the best on
'a9a' in terms of CPU time, while falls behind SA-ADMM in terms of the number of iterations. Therefore, we conclude thatIncrePA works comparable to ADMM-based incremental methods and is much better than PA-based PA-ASGD method.

Before proceeding to the nonconvex composite penalty experiments, we would like to point out that, for the convex composite penalty, as a PA method, the proposed method has generally better performance than stochastic gradient-based method: PA-ASGD, in terms of all performance metrics we have tried so far. As an incremental gradient-based method, the proposed method has comparable performance with SDCA-ADMM and SA-ADMM, but the merit of the proposed method is two-fold: (1) The convergence analysis of SDCA-ADMM relies on the local strongly convexity of the loss function. In addition, SDCA-ADMM requires that the dual problem should be in structure for ADMM to be applied to, which causes a stricter problem format and therefore limits its application domain. For example, in the above case, SDCA-ADMM cannot work at all because the dual parts do not fit into the structure for ADMM to be applied to when being put together, despite each dual of their primal correspondences is easy to take. By contrast, the proposed method has given the convergence analysis for both of general convex loss and strongly convex loss problems. Further, the format of objective function in the proposed method is more general than SDCA-ADMM; (2) SA-ADMM lacks convergence analysis for strongly convex loss problem, but the proposed one does.

4.5.3 Experiment 3: Solving nonconvex composite penalty of capped $\ell_1$ overlapping group lasso

This subsection studies the efficiency of IncrePA by comparing it with the other two algorithms, i.e. GD-PAN and CCCP, for such nonconvex composite penalty. In this experiment, we considered capped-$\ell_1$ norm coupled nonconvex overlapping group lasso:

$$\min_{x \in \mathbb{R}^d} \frac{1}{2n} ||y - Sx||_2^2 + \lambda \sum_{k=1}^{K} \min\{||x_{g_k}||, \theta\}. \quad (4.5.34)$$

We used a synthetic data generated in the same way as \textbf{[146]}. Specifically, the data $s_i$
Figure 4.5: Nonconvex composite penalty: Empirical risk on training data versus CPU time of least square loss with capped $\ell_1$ norm overlapping group lasso on synthetic datasets.

was generated independently and identically distributed from the normal distribution $\mathcal{N}(0, 1)$. The ground truth parameter $x^*$ was generated as $x_j^* = (-1)^j \exp(-\frac{j-1}{100})$. Therefore the dimension was $d = 90K + 10$ features. We set $y_i = (x^*)_T s_i + \vartheta_i$, where $\vartheta_i = 10 \mathcal{N}(0, 1)$. We used the following pairs of $(K, N)$ with both growing dimension and data number: $(5, 500), (10, 1000), (20, 2000), (30, 3000), (30, 5000), (30, 6000)$. We also fixed the dimension to $K = 30$ and increased the data number with $N = 4000, 5000, 6000, 8000$. We compared the proposed algorithm with GD_PAN and CCCP. For the GD_PAN method, we used the step size as suggested in [159]. We fixed the parameters for all different methods to be $(\lambda, \theta) = (K/10, 0.1)$. For the proposed method, we chose the step size to obtain the largest descent in one pass over 5% of the data as suggested in [91]. We ran each algorithm 10 times. The performance of the three algorithms is shown in Figure 4.5 by plotting the objective value over CPU time. When the dataset is small, e.g. $(K, N) = (5, 500), (10, 1000)$, GD_PAN actually works better than the proposed method. However, when the dataset becomes large, the proposed method is much better, which indicates that the proposed method has better scalability.
Figure 4.6: Empirical risk on training data versus effective passes with Nonconvex Graph-guided Lasso on four real datasets.

4.5.4 Experiment 4: Solving nonconvex composite penalty of capped $\ell_1$ graph-guided lasso

In this subsection, we considered nonconvex composite penalty by implementing the capped $\ell_1$-norm with graph-guided lasso penalty on four real datasets (see Table 4.1). The graph is again constructed by sparse inverse covariance matrix. Again, we compared the proposed algorithm with GD-PAN and CCCP. We reported the training efficiency in terms of training loss (objective value) over the effective pass of data in Figure 4.6. It can be seen that the proposed method is consistently better than GD-PAN and CCCP in terms of training. For the ‘20 newsgroup’, ‘a9a’ and ‘protein’ datasets, the proposed method is much faster than the other two methods, while these three methods perform closely on the ‘covtype’. We also reported test loss over the effective passes to show the generalization performance of the learned variable.
Figure 4.7: Test loss on testing data versus effective passes with Nonconvex Graph-guided Lasso on four real datasets.

in Figure 4.7. It can be seen that the proposed algorithm is more advantageous compared with both GD-PAN and CCCP on ‘20 newsgroup’, ‘a9a’ and ‘protein’ datasets, while all of them perform similar on ‘covtype’ dataset. To sum up, the proposed method is more efficient than GD-PAN and CCCP in both training and testing.

4.6 Summary

In this chapter, we have proposed a new incremental gradient method for empirical risk minimization regularized by composite regularizer. As a PA technique-based method, it is more efficient and faster than its existing batch and stochastic counterpart. When applied to convex composite penalties, compared with popular ADMM-based incremental gradient, it has comparable performance, yet enjoys more compact
update form and simpler theoretical analysis by virtue of the PA technique. Also, we have proposed a variant for nonconvex composite penalties, which has better scalability than the existing PA-based methods. Experimental results on four real datasets have shown its efficiency and satisfactory generalization performance for convex composite penalties. Further, experiments on both synthetic and real datasets has demonstrated its better scalability and improved efficiency for the nonconvex composite penalties.
Chapter 5

Scalable Spectral k-Support Norm Regularization for Robust low-rank Subspace Learning

5.1 Introduction

Recovering low rank matrix from gross corruptions has been a fundamental problem in machine learning, data mining and computer vision. Representative applications include collaborative filtering [136], background modeling [16], face clustering [86], among others. The gross corruption, also known as outliers, is often modeled by a sparse noise matrix. The robust low rank subspace learning tasks then aim to learn the low rank matrix with simultaneously minimizing the sparse noise matrix. In general, the low rank matrix and sparse matrix are required to satisfy certain linear constraints. With different designs of linear map, various tasks can be formulated by this linear constraint joint low rank and sparse matrix minimization problem, including robust principal component analysis (RPCA) [16] and low rank representation (LRR) [86].

Regarding the NP-hard rank minimization, nuclear norm is the most popular convex relaxation. As pointed out by [36], nuclear norm is actually the tightest convex relaxation of the nonconvex cardinality function (i.e. $\ell_0$ norm function) of
its singular values \cite{36} under unit infinite norm ball. Recently, k-support norm \cite{3}, which seeks the tightest convex relaxation of the $\ell_0$ norm (being value k) under unit $\ell_2$-norm ball rather than infinite norm ball, has been studied. It has been shown that k-support norm outperforms the other convex relaxations such as $\ell_1$ norm \cite{131} and elastic net \cite{161} for sparsity estimation, both theoretically and practically. Motivated by the success of k-support norm, spectral k-support norm \cite{36,35,95} has been proposed to prompt low rankness of matrix by applying the k-support norm to the singular values of the matrix. Compared with nuclear norm, it provides tight relaxation of the rank k matrices under unit $\ell_2$ norm ball of its singular values rather than infinite norm ball, which is often more preferred \cite{36,95}. Papers \cite{95} and \cite{96} have studied the spectral k-support norm in low rank matrix completion task and have reported the performance against the other convex penalties. \cite{95} also shows the link of the spectral k-support norm between cluster norm used in the multi-task learning context. Furthermore, \cite{96} extends it to spectral $(k,p)$-support norm to capture the decay of singular values of the underlying low rank matrix. Despite the superior recovery performance compared with other convex relaxations like nuclear norm, the spectral k-support norm is much more difficult to be optimized, which therefore severely limits its application domain, particularly for big data analysis. Although methods developed for k-support norm that relies on proximal map of the squared k-support norm \cite{3,36,80} can be migrated to spectral k-support norm, its computation is laborious. A major reason is the full SVD decomposition involved in the proximal mapping computation. Furthermore, restricted by the property of the k-support norm, efficient approximation methods for nuclear norm (e.g. power method and Lanczos method) that requires leading singular values only are hardly applicable to spectral k-support norm. Further, a search operation that segments singular values into certain groups also needs additional computation.

In this chapter, we will study the spectral k-support norm for robust low rank subspace learning task. Regarding optimization, it is apparently more challenging to design an efficient and scalable algorithm than the previous research focusing on
matrix completion \cite{95,96}, given the additional linear constraint. We propose two variants for utilizing the spectral k-support norm, of which one uses the squared form as previous methods do. In the other variant, we show that we can also directly design an optimization algorithm for the original spectral k-support norm, which is hardly possible for most existing proximal gradient-based methods. We first follow the common practice to get rid of the linear constraint by Lagrangian dual. Next, instead of directly optimizing the Lagrangian dual alternatively as common ADM-based methods do, we further convert the problem by Fenchel conjugation \cite{149}. The optimization of the resultant dual objective can then be solved via accelerated proximal gradient method (APG) \cite{101}, which only requires to evaluate the polar operator of spectral k-support norm, plus the proximal mapping related to infinity norm. Both of them are more computational efficient than the proximal map related to spectral k-support norm, in which the per-iteration cost is linear instead of superlinear. In principal, we follow the recently proposed primal-dual framework \cite{102,149} and recover the primal low rank variable along the dual APG iterations \cite{149} By studying the (sub)gradient set of the loss function of the dual objective, we also incorporate the line-search strategy \cite{149} that can adapt to the smoothness of the dual objective in the sense of Hölder continuity. Also, please note that line-search is possible in our method because the dual norm of the spectral k-support norm is more efficient to compute than itself, which is another advantage brought about by our dual conversion. Per-iteration complexity analysis shows that the time complexity of our method is linear with respect to the size of low rank matrix, whereas ADM-based methods would involve super-linear complexity.

In summary, we propose a polar operator-based algorithm featuring the following merits:

1. The proposed algorithm costs only linear per-iteration complexity rather than super-linear if proximal ADM method is adopted;

2. Our method is flexible to deal with both squared k-spectral norm and itself, whereas most of previous methods are confined with the former form; Also, our
method is general so that it can be adapted to various choices of linear map, constant matrix and sparse norm to suit different model;

3. Our method converts to an equivalent dual form that deals with the dual spectral k-support norm, which is easier to compute than the primal norm. This further enables us to incorporate a line-search strategy to adapt to the degree and constant of the smoothness of the dual objective in the sense of Hölder smoothness.

5.2 Preliminary: Scalable Algorithm with Spectral k-Support Norm

With the spectral k-support norm, the robust low rank subspace learning problem in eq.(2.3.18), featuring a nonsmooth and linear constraint optimization problem, is difficult to be solved in a scalable way, which severely limits the application of the spectral k-support norm from the practical perspective. In this subsection, we will explain that popular approaches to scaling nuclear norm regularization under this model is not applicable to spectral k-support norm. Specially, for nuclear norm, matrix factorization-based methods and ADM-type methods are both effective algorithms for solving eq.(2.3.18) efficiently, but none of them can be applied to solve spectral k-support norm regularized problem efficiently. The matrix factorization-based methods crucially rely on the following property of nuclear norm: $||Z||_* = \min_{P,Q} \frac{1}{2} ||P||_F^2 + \frac{1}{2} ||Q||_F^2, \ s.t. \ Z = PQ$, which is not applicable to spectral k-support norm. For the ADM-type methods, we argue that the proximal operator-based ADM method and its variants cannot optimize the spectral k-support norm regularized robust subspace learning problem in a scalable way by briefly deriving such an algorithm based on a particular linearized ADMM scheme \cite{55} as follows:

$$ ||L||_{sp,k} + \lambda ||S||_1 + \langle \Gamma, BL + S - M \rangle + \frac{\rho}{2} ||BL + S - BM||_F^2. \quad (5.2.1) $$
Then, it will update $L, S, \Gamma$ in an alternate fashion. In particular, to optimize $L$, we linearize the squared Frobenius norm term

$$\arg \min_L ||L||_{sp,k} + \langle B^T \Gamma_t, L \rangle + \rho B^T (BL_t + S_t - BM, L) \rangle + \frac{\eta}{2} ||L - L_t||_F^2.$$  

(5.2.2)

This will require the proximal operator related to $|| \cdot ||_{sp,k}$, 

$$L_{t+1} = \arg \min_L ||L||_{sp,k}^2 + \frac{\eta}{2} ||L - C_t||_F^2.$$ 

(5.2.3)

Please note that eq.(5.2.3) uses the squared spectral k-support norm instead, which has yet to know whether a closed-form solution exists for this norm in the literature. Actually, all existing methods resort to the squared k-support norm, which has closed-form solution. It is not difficult to adapt the proximal operators for the squared k-support norm \cite{3, 80, 36} for spectral k-support norm. However, all existing proximal mappings cannot be computed in a scalable way. The main bottleneck is that proximal mapping would require a full SVD decomposition plus a searching step to segment the singular values into three different groups for different types of computation. \cite{80} improves upon \cite{3} by using binary search instead of the exhaustive search, and \cite{36} proposes to solve the proximal mapping of the spectral k-support norm by computing the proximal mapping of its dual norm. However, none of these methods are able to avoid the full SVD because the search step and the subsequent computation both rely on all of the singular values. Nuclear norm-based ADM method is able to avoid such full SVD by an approximation technique that only requires to compute a few leading singular values, which is, unfortunately, not applicable here for spectral k-support norm. As a result, such ADM-based method would incur super-linear per-iteration cost that severely limits the scalability of spectral k-support norm’s utilization under this model.

### 5.3 The Proposed Method

In this section, we present our proposed method for learning robust low rank subspace with spectral k-support norm regularization in an efficient way. We begin with two
reformulations and derive the corresponding equivalent problem based on Fenchel dual, one of which uses the squared spectral k-support norm and the other uses the original spectral k-support norm. The reformulated equivalent problems, referred as dual objectives, allow more efficient computation, in which the per-iteration cost hinges on solving a linear subproblem, referred as linear oracle evaluation of the spectral k-support norm. The linear oracle evaluation only needs to compute the leading k-singular value decomposition (SVD), avoiding the full SVD computation otherwise required by proximal mapping-based ADM methods, is known to be more efficient to compute, especially with Lanczos method or power method techniques. Also, our method does not require the search step of the proximal mapping of spectral k-support norm. In addition, we study the smoothness of our loss function of the dual objective and incorporate a line-search strategy that can adapt to the smoothness change in the sense of Hölder continuity to further accelerate the algorithm.

5.3.1 Formulation I: Usage with Squared Spectral k-Support Norm

In our first formulation, we utilize the squared spectral k-support norm, which is adopted by almost all proximal mapping-based methods [3, 80, 36]. Let $L$ denote the target low rank variable, we are solving the following constraint form of robust low rank subspace model:

$$
\min_L \frac{1}{2} ||L||_{sp,k}^2, \ s.t. \ ||S||_s \leq \tau, \ \mathcal{B}(M - L) = S.
$$

(5.3.4)

The above formulation amounts to the constraint $||\mathcal{B}(M - L)||_s$, which is considered more natural than regularization formulation because it directly signifies the tolerance on the misfit [2]. With a proper choice of $\tau$, it is equivalent to the regularized form in eq.(2.3.18). Denoting the dual variable by $\Gamma$, by using the Lagrangian dual to handle the linear constraint $\mathcal{B}(M - L) = S$, we also get the following Lagrangian formulation,

$$
\max_{\Gamma} \ \min_{s,L,||S||_s \leq \tau} \left[ \frac{1}{2} ||L||_{sp,k}^2 + \langle \Gamma, BL + S - BM \rangle \right].
$$

(5.3.5)
However, instead of performing alternative updating strategy which would incur the usage of the expensive proximal map of the square form of the spectral k-support norm, we further convert eq. (5.3.5) by Fenchel conjugation, as summarized in the following proposition.

**Proposition 5.3.1.** To solve the maximization problem in eq. (5.3.5), it is equivalent to solve the following minimization problem w.r.t the Lagrangian dual variable $\Gamma$,

$$
\min_{\Gamma} f(\Gamma) + r(\Gamma), \text{ where } \tag{5.3.6}
$$

$$
f(\Gamma) = \frac{1}{2} (|| - B^T \Gamma ||_{sp,k}^*)^2 + \langle \Gamma, BM \rangle, \tag{5.3.7}
$$

$$
r(\Gamma) = \tau || - \Gamma ||_s^*. \tag{5.3.8}
$$

In eq. (5.3.8), $|| \cdot ||_s^*$ denotes the dual norm of $|| \cdot ||_s$, e.g. $|| \cdot ||_\infty$ for $|| \cdot ||_1$ norm and $|| \cdot ||_{2,\infty}$ for $|| \cdot ||_{2,1}$ norm. Proposition 5.3.1 converts the optimization of eq. (5.3.5) to eq. (5.3.6) that is referred as dual objective in the sequel. To solve eq. (5.3.5) with respect to Lagrangian dual variable $\Gamma$, we can apply the proximal gradient descent algorithm [101]. The proximal map is now related to $|| \cdot ||_s^*$, which is essentially equivalent to projection onto $|| \cdot ||_s^*$ unit ball and is not expensive [30, 133]. Hence another major per-iteration cost would be the gradient evaluation of $f(\Gamma)$. Before proceeding to the computation of the gradient, we give a brief proof of Proposition 5.3.1 which would reveal a particular choice of (sub)gradient of the loss function $f(\Gamma)$.

**Proof.** To prove Proposition 5.3.1, we begin with the following sequence of equivalence
relations:

\[
\max_{\Gamma} \min_{S,L,||S||_s \leq \tau} \left[ \frac{1}{2} ||L||^2_{sp,k} + \langle \Gamma, BL + S - BM \rangle \right] \\
\iff \max_{\Gamma} \left[ \min_{L} \left( \frac{1}{2} ||L||^2_{sp,k} + \langle \Gamma, BL \rangle \right) - \langle \Gamma, BM \rangle \\
+ \min_{||S||_s \leq \tau} \left( \langle \Gamma, S \rangle \right) \right] \\
\iff \max_{\Gamma} \left[ \min_{L} - \left( \langle -B^T \Gamma, L \rangle - \frac{1}{2} ||L||^2_{sp,k} \right) - \langle \Gamma, BM \rangle \\
+ \min_{||S||_s \leq \tau} \left( -\langle -\Gamma, S \rangle \right) \right] \\
\iff \max_{\Gamma} - \left[ \max_{L} \left( \langle -B^T \Gamma, L \rangle - \frac{1}{2} ||L||^2_{sp,k} \right) + \langle \Gamma, BM \rangle \\
+ \max_{||S||_s \leq \tau} \left( -\langle -\Gamma, S \rangle \right) \right].
\] (5.3.9)

The first and the second term in the square bracket can be combined and converted as follows:

\[
\max_{L} \left( \langle -B^T \Gamma, L \rangle - \frac{1}{2} ||L||^2_{sp,k} \right) + \langle \Gamma, BM \rangle \\
= \frac{1}{2} \left( || -B^T \Gamma||^2_{sp,k} \right) + \langle \Gamma, BM \rangle := f(\Gamma)
\] (5.3.10)

The third term in the square bracket can be rewritten based on the definition of dual norm of \( || \cdot ||_s \), i.e.

\[
\max_{||S||_s \leq \tau} \langle -\Gamma, S \rangle = \max_{||S/\tau||_s \leq 1} \langle -\tau \Gamma, S/\tau \rangle = \tau || -\Gamma ||^*_s := r(\Gamma).
\] (5.3.11)

By combining the above derivation together, we can solve the right-hand side problem to equivalently solve the original Lagrangian dual problem on the left-hand side of the following equation:

\[
\max_{\Gamma} \min_{s,L,||S||_s \leq \tau} \left[ \frac{1}{2} ||L||^2_{sp,k} + \langle \Gamma, BL + S - BM \rangle \right] \\
\iff -\min_{\Gamma} \left( f(\Gamma) + r(\Gamma) \right).
\] (5.3.12) (5.3.13)

\[\Box\]

Based on eq.\([5.3.10]\) (i.e. taking the derivative of the first line in eq.\([5.3.10]\) w.r.t. \(\Gamma\)), we have a particular choice of the (sub)gradient of the dual loss function \(f(\Gamma)\), as shown in the following corollary.
Corollary 5.3.2. Denote a particular subgradient of \( \partial f(\Gamma) \) by \( g(\Gamma) \), then it can be computed as

\[
g(\Gamma) = -B L^# + B M, \text{ where } L^# = \arg \max_{||A||_{sp,k} \leq 1} \langle -B^T \Gamma, A \rangle.
\] (5.3.14)

According to Proposition (5.3.2), the computation of computing the (sub)gradient of the dual objective comes from computing \( L^# \), which requires to solve a linear problem \( \arg \max_{||A||_{sp,k} \leq 1} \langle -B^T \Gamma, A \rangle \). The spectral k-support norm, as an gauge function \[\|\cdot\|\] (i.e. nonnegative, positively homogeneous convex functions vanishing at the origin), allows the linear subproblem to be equivalently solved by the following polar operator:

\[
\arg \max_{A \in \mathcal{A}} \langle -B^T \Gamma, A \rangle.
\] (5.3.15)

Recall that \( \mathcal{A} \) is the set of “atoms” of the spectral k-support norm defined in eq.(2.3.26) and also note that the structure of the \( A \in \mathcal{A} \) constraint set is much simpler to deal with than \( ||A||_{sp,k} \leq 1 \). In fact, the polar operator has closed-form solution which only computes top k-SVD of matrix \( B^T \Gamma \) in eq.(5.3.14), as shown in the following lemma from [96]:

Lemma 5.3.3. Denote a particular SVD of an arbitrary matrix \( X \in \mathbb{R}^{(m,n)} \) by \( X = U \text{diag}(\sigma)V^T \). Then the polar operator of the spectral \( (k,p) \)-support norm, i.e. \( L^# = \arg \sup_{A \in \mathcal{A}} \langle X, A \rangle \) (recall that \( \mathcal{A} \) is the “atomic” set in eq.(2.3.26)), admits the closed-form solution as \( L^# = U \text{diag}(s)V^T \), where

\[
s_i = \begin{cases} 
\frac{\sigma_i}{||\sigma||_{sp,(k,p)}}, & i = 1, \ldots, k \\
0, & i = k + 1, \ldots \min\{m,n\}.
\end{cases}
\] (5.3.16)

Recall that \( ||\sigma||_{sp,(k,p)} \) is the dual spectral \( (k,p) \) support norm of \( X \) in eq.(2.3.28) and simply set \( p = q = 2 \) for spectral k-support norm. According to Lemma (5.3.3), the computation of the polar operator, and thus the gradient of the dual objective, only involves the top k-SVD, which is more efficient to evaluate than full SVD, especially with Lanczos [82] or perhaps power method [16] techniques. Please note that although [96] also utilizes the polar operator, their methods are based on vanilla
Frank-Wolfe algorithm, which is not applicable when additional linear constraint is involved.

5.3.2 Formulation II: Usage with Spectral k-Support Norm

In this subsection, we propose our second formulation that utilizes the spectral k-support norm itself, which is impossible for proximal mapping-based approach due to the lack of known closed-form proximal mapping. Again, we begin with the following constraint formulation:

\[
\min_L ||L||_{sp,k}, \text{ s.t. } ||S||_s \leq \tau, B(M - L) = S.
\]  

(5.3.17)

Before converting it to Lagrangian dual form to get rid of the equality constraint, we introduce an auxiliary variable \(v_l\) with:

\[
\min_{v_l} v_l, \text{ s.t. } ||S||_1 \leq \tau, B(M - L) = S, ||L||_{sp,k} \leq v_l \leq Q_l,
\]  

(5.3.18)

where \(Q_l\) is a constant estimation of the upper bound of \(||L||_{sp,k}\). This technique has been previously introduced by [48] and later also adopted by [99] for extending Frank-Wolfe algorithms [60] to norm regularization problem. Again, denoting the Lagrangian dual variable by \(\Gamma\), we have

\[
\max_{\Gamma} \min_{L,v_l,S} [v_l + \langle \Gamma, BL + S - BM \rangle] \big||L||_{sp,k} \leq v_l \leq Q_l, ||S||_s \leq \tau\].
\]  

(5.3.19)

We then further transform the above formulation by Fenchel conjugation summarized by the following proposition.

**Proposition 5.3.4.** To solve the maximization problem in eq. (5.3.19), it is equivalent to solve the following minimization problem with respect to the Lagrangian dual variable \(\Gamma\):

\[
\min_{\Gamma} f(\Gamma) + r(\Gamma), \text{ where } \]

\[
f(\Gamma) = \max\{0, (Q_l|| - B^T\Gamma||_{sp,k} - 1)\} + \langle \Gamma, BM \rangle,
\]  

(5.3.20)

\[
r(\Gamma) = \tau|| - \Gamma||_s^*.
\]  

(5.3.22)
Proof. To prove Proposition 5.3.4, we begin with the following equivalent relationship, which is related to the low rank component $L$ and $v_l$:

$$
\min_{v_l, L} \left[ v_l + \langle \Gamma, BL \rangle \right|_{L_{sp,k}} \leq v_l \leq Q_l
$$

$$
= \min_{v_l, A} \left[ v_l \left( 1 + \langle B^T \Gamma, A \rangle \right) \right|_{L_{sp,k}} \leq 1, \ 0 \leq v_l \leq Q_l
$$

$$
= \min_{0 \leq v_l \leq Q_l} \left[ v_l \left( 1 - \max_{A \in A} \langle -B^T \Gamma, A \rangle \right) \right]
$$

$$
= \max_{0 \leq v_l \leq Q_l} \left[ v_l \left( \| - B^T \Gamma \|_{sp,k}^* - 1 \right) \right].
$$

(5.3.23)

If $l(\Gamma) := (\| - B^T \Gamma \|_{sp,k}^* - 1) > 0$, $\max_{0 \leq v_l \leq Q_l} \left[ v_l l(\Gamma) \right] = Q_l l(\Gamma)$ because the optimal $v_l^\# = Q_l$; Otherwise, $\max_{0 \leq v_l \leq Q_l} \left[ v_l l(\Gamma) \right] = 0$ because the optimal $v_l^\# = 0$. That is,

$$
\min_{v_l, L} \left[ v_l + \langle \Gamma, BL \rangle \right|_{L_{sp,k}} \leq v_l \leq Q_l
$$

$$
= \max_{0 \leq v_l \leq Q_l} \left[ v_l \left( \| - B^T \Gamma \|_{sp,k}^* - 1 \right) \right].
$$

(5.3.24)

As for the sparse component $S$, we can obtain the reformulation similar to Formulation I in the previous subsection, i.e.

$$
\min_{\|S\|_s \leq \tau} \langle \Gamma, S \rangle = \max_{\|S\|_s \leq \tau} \langle -\Gamma, S \rangle = -\tau - \tau\| - \Gamma\|_s^* := -r(\Gamma).
$$

(5.3.25)

Combining the above together, we have the following dual problem:

$$
\max_{\Gamma} - \left[ \max \left\{ 0, Q_l l(\Gamma) \right\} + \langle \Gamma, BM \rangle + r(\Gamma) \right]
$$

$$
= - \min_{\Gamma} \left[ \max \left\{ 0, Q_l l(\Gamma) \right\} + \langle \Gamma, BM \rangle + r(\Gamma) \right].
$$

(5.3.26)

Therefore, we can equivalently solve

$$
\min_{\Gamma} \left[ \max \left\{ 0, Q_l l(\Gamma) \right\} + \langle \Gamma, BM \rangle + r(\Gamma) \right] := \min_{\Gamma} f(\Gamma) + r(\Gamma).
$$

(5.3.27)

The next corollary shows a particular choice of (sub)gradient for $f(\Gamma)$.

Corollary 5.3.5. A particular choice of the (sub)gradient for $f(\Gamma)$ is given by $g(\Gamma)$:
\[
g(\Gamma) = \begin{cases} 
BM - Q_tBL^#, & (|| - B^T\Gamma||_{sp,k}^* - 1) > 0 \\
BM - \text{conv}\{0, Q_tBL^#\}, & (|| - B^T\Gamma||_{sp,k}^* - 1) = 0 \\
BM, & (|| - B^T\Gamma||_{sp,k}^* - 1) < 0
\end{cases}
\]

(5.3.28)

where \( L^# = \arg \max_{A \in \mathcal{A}} (-B^T\Gamma, A) \) can be computed according to Lemma 5.3.3.

Corollary 5.3.5 shows that the major computational cost of the (sub)gradient for \( f(\Gamma) \) depends again on the linear optimization problem of evaluating the polar operator of spectral k-support norm. Compared with Formulation I in the previous subsection, to learn with the spectral k-support norm itself, we need to tune one more parameter \( Q_t \), which is used in eq.(5.3.23).

### 5.3.3 Algorithm

**APG for the Dual Objective**

Following [102, 149], we can then solve the converted dual objective with the accelerated proximal gradient descent (APG) [7, 101]. The gradient of each step can be evaluated according to Corollary 5.3.2 and Corollary 5.3.5. In detail, we keep two interpolation sequences \( \hat{\Gamma}_t \) and \( \Gamma_t \), which is typical for APG-type methods. Specifically, in each iteration, the algorithm updates the dual variable \( \Gamma_t \) by,

\[
\Gamma_{t+1} = \arg \min_{\Gamma} f(\hat{\Gamma}_t) + \langle g(\hat{\Gamma}_t), \Gamma - \hat{\Gamma}_t \rangle + \frac{H_{t+1}}{2} ||\Gamma - \hat{\Gamma}_t||^2_F
\]

\[+ r(\Gamma);
\]

\[
\hat{\Gamma}_{t+1} = \Gamma_{t+1} + \frac{\lambda_t - 1}{\lambda_{t+1}} (\Gamma_{t+1} - \Gamma_t),
\]

(5.3.29)

(5.3.30)

where \( \lambda_t \) is a scalar sequence updated iteratively as \( \lambda_{t+1} = \frac{1 + \sqrt{1 + 4\lambda_t^2}}{2} \) with the initial value 1. \( H_t \) is the reciprocal of the step size. Also, recall that \( g(\hat{\Gamma}_t) \) is the gradient of \( f \) at \( \hat{\Gamma}_t \) which can be evaluated by eq.(5.3.14) and eq.(5.3.28).

The subproblem eq.(5.3.29) is actually the proximal mapping related to the dual norm of the sparsity inducing norm, which is essentially to compute the projection.
onto $\ell_1$ norm ball. In detail, eq.(5.3.29) is the proximal mapping corresponds to $\| \cdot \|_s^*$ that is denoted as $\text{prox}_{H_{t+1}}(\hat{\Gamma}_t - g(\hat{\Gamma}_t)/H_{t+1})$,

$$\Gamma_{t+1} = \arg \min_{\Gamma} \frac{1}{2} \| \Gamma - (\hat{\Gamma}_t - g(\hat{\Gamma}_t)/H_{t+1}) \|_F^2 + \frac{\tau}{H_{t+1}} \| \Gamma \|_s^*,$$  

(5.3.31)

which can be equivalently evaluated by the projection on the unit $\| \cdot \|_s$ norm ball according to

$$\Gamma_{t+1} = (\hat{\Gamma}_t - g(\hat{\Gamma}_t)/H_{t+1}) - \frac{\tau}{H_{t+1}} \text{proj} \left( \frac{1}{\tau} (\hat{\Gamma}_t - g(\hat{\Gamma}_t)/H_{t+1}) \right),$$  

(5.3.32)

where $\text{proj}(X)$ denotes the projection operation, e.g. projects onto $\ell_1$-ball or $\ell_2$-ball, both of which allow efficient computation that costs linear complexity with respect to the size of $X$, i.e. $O(mn)$ for $X \in \mathbb{R}^{(m,n)}$.

**Line-search**

In the following, we study the (sub)gradient set of $f(\Gamma)$, which apparently depends on the structure of the (sub)gradient of the dual norm $\| \cdot \|_{sp,k}^*$ (see eq.(5.3.7) and eq.(5.3.21)). To keep the study more general, the following lemma shows the form of (sub)gradient of the dual norm of spectral $(k,p)$-norm $\| \cdot \|_{sp(k,p)}^*$, which is generalized from Proposition 5 in [29] and also see [135].

**Proposition 5.3.6.** For $\Gamma \neq 0$, denote a particular singular value decomposition of $\Gamma$ by $\Gamma = U \text{diag}(\sigma)V^T$ and suppose the singular values satisfies $\sigma_1 \geq \sigma_2 \geq \ldots > \sigma_{k-a+1} = \ldots = \sigma_k = \ldots = \sigma_{k+b} > \ldots \geq \sigma_d$. $q$ satisfies $\frac{1}{p} + \frac{1}{q} = 1$. Then, the subgradient set of the dual norm of the spectral $(k,p)$-support norm at $\Gamma$ is

$$\frac{1}{\| \Gamma \|_{sp(k,p)}^{(q-1)}} \left\{ U[:1:k-a] \text{diag}(\sigma_{1:k-a}^{q-1})V_{[,1:k-a]}^T + U[:k-a+1:k+b] RV_{[,k-b+1:k+b]}^T \right\},$$  

(5.3.33)

where $R$ is a symmetric matrix and satisfies $\| R \|_2 \leq 1$ and $\| R \|_* = a$. In particular, it is differentiable when $\sigma_k > \sigma_{k+1}$ or $\sigma_k = 0$ with the gradient equal to

$$\frac{1}{\| \Gamma \|_{sp(k,p)}^{(q-1)}} \left\{ U[:1:k] \text{diag}(\sigma_{[1:k]}^{q-1})V_{[,1:k]}^T \right\}. $$  

(5.3.34)
According to Proposition (C.3.1), we actually choose eq.(C.3.36) as the (sub)gradient in computing the gradient of $g(\Gamma)$. The conditions of the uniqueness of the subgradient set, i.e. whether $\sigma_k > \sigma_{k+1}$ or $\sigma_k = 0$ is satisfied, can be interpreted as whether the first $k$ singular values of $\Gamma$ are well-separated with the remaining singular values. Proposition (C.3.1) indicates that, when the first $k$ singular values are well-separated, $g(\Gamma)$ would be differentiable. In practice, initializing with a low rank matrix $\Gamma$ (e.g. all-zero matrix), we would expect that the singular values of $\Gamma$ change from satisfying the uniqueness condition (e.g. $\sigma_k = 0$) to dissatisfying across iterations.

Therefore, the smoothness of the dual objective loss $g(\Gamma)$ would change from differentiable to subdifferentiable across iterations, which corresponds to degree $\nu = 1$ to degree $\nu = 0$ in the sense of Hölder continuity, which guarantees the following relationship (for more detailed properties, please see [102]), $||\nabla f(x) - \nabla f(y)|| \leq H_\nu ||x - y||^\nu$, $\forall x, y$, where $\nu \in [0, 1]$ is referred as the degree of smoothness and $H_\nu$ is assumed finite that is defined by

$$H_\nu := H_\nu(f) = \sup_{x \neq y \in \mathcal{D}} \frac{||\nabla f(x) - \nabla f(y)||}{||x - y||^{\nu}}. \quad (5.3.35)$$

Next, we utilize a line-search scheme proposed recently by [149], which is able to automatically adapt to both the degree and constant of the Hölder continuity of the dual objective and thus chooses more optimal step size. We denote the reciprocal of the step size at iteration $t$ by $H_t$. According to proximal gradient update related to $r(\Gamma)$, we have

$$Q_{H_t}(\Gamma; \hat{\Gamma}_t) = f(\hat{\Gamma}_t) + \langle g(\hat{\Gamma}_t), \Gamma - \hat{\Gamma}_t \rangle + \frac{H_{t+1}}{2} ||\Gamma - \hat{\Gamma}_t||_F^2. \quad (5.3.36)$$

In essence, the line search aims to find the minimum $H_{t+1}$ (corresponding to the largest step size) that satisfies the following criterion:

$$f(\Gamma_{t+1}) \leq Q_{H_{t+1}}(\Gamma_{t+1}; \hat{\Gamma}_t) + \frac{\epsilon}{2\lambda_t}, \quad (5.3.37)$$

where $\epsilon$ is the error tolerance and $\lambda_t$ is the sequence kept by APG algorithm.
Primal Variable Recovery

Thus far, we have dealt with the dual objective and dual variable. However, our ultimate goal is the primal variable $L$. To do so, we follow [149] to simultaneously maintain the primal variable sequence $L_t$ across dual variable updating procedure, i.e. $L_{t+1} = (1 - \gamma_t)L_t + \gamma_t L^\#$, where $\gamma_t$ is the weighting parameter and $L^\#$ is the polar operator result computed during the gradient evaluation. According to [149], $\gamma_t$ is constructed by also taking information from the adaptive step size

$$\gamma_t = \frac{\lambda_t / H_t}{\sum_{i=1}^{t} \lambda_i / H_i}.$$  

(5.3.38)

This primal update step is similar to the Frank-Wolfe algorithm. With a constant step size, the weighting strategy would look even more similar to the “standard” Frank-Wolfe weighting strategy $\frac{2}{t+1}$. By combining the above parts, the complete procedure is summarized in Algorithm 12.

Algorithm 12 Proposed algorithm

\textbf{Input:} $\Gamma_0$, $\hat{\Gamma}_0$, $L_0 = \mathbf{0}_{(m,n)}$, $\lambda_0 = 1$, $v_t$, $Q_t$, $\tau$, $\epsilon > 0$, $t_{\text{max}}$;

1: \textbf{for} $t = 0, 1, \ldots, t_{\text{max}}$ \textbf{do}
2: \hspace{1em} Compute $L^\#$ by evaluating the polar operator in eq. (5.3.15) at $\hat{\Gamma}_t$;
3: \hspace{1em} Compute the (sub)gradient $g(\hat{\Gamma}_t)$ of $f(\Gamma)$ at $\hat{\Gamma}_t$ by eq. (5.3.14) (for Formulation I) or eq. (5.3.28) (for Formulation II);
4: \hspace{1em} Compute $\Gamma_{t+1} = \text{prox}_{H_t^{-1}r(\Gamma)}(\hat{\Gamma}_t - H_t^{-1}g(\hat{\Gamma}_t))$ by eq. (5.3.32), where $H_{t+1}$ is decided by line-search subroutine: \textbf{line-search}($\hat{\Gamma}_t, g(\hat{\Gamma}_t), H_t, \epsilon, \lambda_t$);
5: \hspace{1em} Update the weight $\gamma_t$ for primal recovery by eq. (5.3.38);
6: \hspace{1em} Update the sequence $\lambda_{t+1}$: $\lambda_{t+1} = \frac{1+\sqrt{1+4\lambda_t^2}}{2}$;
7: \hspace{1em} Update interpolation sequence $\hat{\Gamma}_{t+1} = \Gamma_{t+1} + \frac{\lambda_{t+1}-1}{\lambda_{t+1}}(\Gamma_{t+1} - \Gamma_t)$;
8: \hspace{1em} Update the primal sequence $L_{t+1} = (1 - \gamma_t)L_t + \gamma_t L^\#$.
9: \textbf{end for}
10: \textbf{Return:} $L_{t_{\text{max}}}$;

The following is the line-search subroutine.
Algorithm 13 line-search subroutine

Input: $\hat{\Gamma}, g(\hat{\Gamma}), H_0, \epsilon, \lambda$

1: for $i = 0, 1, \ldots, i_{\max}$ do
2: $\Gamma_{i+1} = prox_{H^{-1}_{i} r(\Gamma)}(\hat{\Gamma} - H^{-1}_{i} g(\hat{\Gamma}))$
3: if $f(\Gamma_{i+1}) \leq f(\hat{\Gamma}) + \langle g(\hat{\Gamma}), \Gamma_{i+1} - \hat{\Gamma} \rangle + \frac{H_{i}}{2} \| \Gamma_{i+1} - \hat{\Gamma} \|_F^2 + \frac{\epsilon}{2\lambda}$ then
4: break;
5: else
6: $H_{i+1} = 2H_{i}$
7: end if
8: end for
9: Return: $\Gamma_i, H_i$

Algorithm Analysis

To recover an underlying low rank matrix of size $(m, n)$, the time complexity of each part of Algorithm 12 is as follows: step 2 costs $O(kmn)$ to compute the top $k$ SVD; step 3 is simply the point-wise multiplication and summation, which costs $O(mn)$; the proximal map of $r(\Gamma)$ in step 4 takes $O(mn)$ which mainly comes from projection onto sparse norm ball $133$; the line search in step 4 costs $O(i_{\max}kmn)$ to compute at most $i_{\max}$ times dual loss value that requires top $k$ SVD; step 8 costs $O(mn)$. Therefore, the per-iteration complexity is $O(i_{\max}kmn)$, where $i_{\max}$ is 2 on average as observed by 149. Recall that proximal map-based ADM methods would cost $O(\min\{m, n\}mn)$ to compute the full SVD and $\min\{m, n\}log(\min\{m, n\})$ to compute the proximal map of the singular values of the target matrix 95. For practical applications, $k$ is often much smaller than $\min\{m, n\}$, e.g. we set $k=3$ for tasks in subsection (4.2), (4.3). Hence, the proposed method enjoys much lower per-iteration cost.

Now we discuss the convergence behavior of the proposed method by Theorem 2 from 149, depicted by the following theorem.

Theorem 5.3.7. The primal sequence $L_t$ generated by Algorithm 1 converges with
the worst case iteration number to achieve $\epsilon$ error with $t_{max} = O(\inf_{\nu \in [0,1]}(\frac{H_{\nu}}{\epsilon})^{\frac{2}{\nu}})$.

With the smooth objective, i.e. $\nu = 1$, the worst-case iteration number is the same as the one of Frank-Wolfe type algorithms that trade off lower per-iteration complexity with slower convergence rate to scale to larger problem. Also, in practice we find the line-search condition is too conservative. Actually, more efficient implementation can be made by checking the line-search condition every 5 to 10 iterations instead of one per-iteration.

### 5.4 Experiment

In this section, we study the empirical performance of the proposed method on both synthetic and real datasets to test on the RPCA model in eq.(2.3.19). In our implementation, we solve the k-SVD by the lansvd function in the PROPACK package. We empirically set k to be equal or slightly larger than the desired rank of the low rank matrix, which can also be selected by cross-validation. All experiments are done on a laptop computer running MATLAB.

We compare with 1) IALM uses nuclear norm as low rank penalty; 2) PSSV uses partial sum of singular values, i.e. omits the leading singular values in the nuclear norm, which is nonconvex; 3) FWT also uses nuclear norm, but it is an FW-based method instead of proximal mapping. We use recommended or default parameter settings for these compared methods. We do not compare with neither Reg$\ell_1$-ALM which imposes additional assumption that $L = PZ$, where $P$ is orthogonal and $Z$ is low rank, nor the composition of nuclear norm with nonconvex functions like SCAD and MCP functions. For the former, we can expect performance gain if we substitute the nuclear norm penalty with spectral k-support norm for the corresponding low rank part. For the latter, we omit them because this chapter focuses only on studying whether the spectral k-support can
be a better and computational feasible convex relaxation than nuclear norm for the robust subspace learning problems.

Also, we would like to point out that our algorithm can actually be applied to more general joint low rank and sparse minimization model by taking different linear map and constant matrix in eq.(2.3.18). A representative problem is the low rank representation problem (LRR) [86], where $M$ can be identity matrix $I$ and $B$ equals input data matrix $D$. With $\ell_{2,1}$ norm [147] to promote column-wise sparsity, the problem becomes $\text{arg \ min}_{L,S} \|L\|_{sp,k} + \|S\|_{2,1}$, s.t. $D - DL = S$. In this regard, the proposed method is more favorable than algorithms dedicated only to RPCA problem.

5.4.1 Synthetic Data

This subsection evaluates the performance of the proposed algorithm on synthetic data. We generated the ground truth $d \times d$ low rank matrix $G$ by first generating random matrix uniformly sampled within 0 to 1, which was then truncated by $\text{lansvd}$ with the various rank ratio $r$. We added random Gaussian noise $\mathcal{N}(0,0.1)$ to $G$. Finally, we obtained the input matrix $M$ for testing by randomly setting matrix elements to either -20 or +20 in $G$ with a series of corruption ratio $c$, which are outliers.

Figure 5.1 reports the recovery performance under a series degree of corruption. We varied the corruption percentage from 1% to 10%. The data dimension was fixed with $1000 \times 1000$ and the rank ratio was set at 10%. We measured the reconstruction performance by $\ell_1$ relative error $\frac{\|L-G\|_1}{d \times d}$ (left subplot) and $\ell_2$ relative error $\frac{\|L-G\|_2}{\|G\|_F}$ (right subplot), where $G$ is the ground truth matrix and $L$ is the output of algorithms. From Figure 5.1 it can be seen that the recovery error increases with the more outliers. The two formulations of the spectral k-support norm regularized RPCA algorithm perform closely and are better than proximal mapping-based nuclear norm regularized (solved by IALM) and partial singular value sum regularizer algorithms (solved by PSSV). Note that the performance of PSSV is close to that of IALM, both
of which are slightly worse. Among these algorithms, the performance of the FWT method is the worst. Although it uses proximal step for the sparse matrix update, the low rank part is still updated by pure Frank-Wolfe strategy, which is slow and cannot obtain enough decease of the objective compared to proximal algorithm for either primal (like ALM/ADMM) or for the dual form without further local refinement [60].

In Figure 5.2, we compared the algorithms with rank ratio varying from 1% to 10%, while fixing corruption ratio to 5% and data dimension to 1000 × 1000. Recovery performance under $\ell_1$ (left) and $\ell_2$ norm (right) are reported. Again, the proposed method with two formulations performs closely and are better than the counterparts. Therefore, the spectral k-support norm is superior to nuclear norm for RPCA task in terms of recovering performance. Furthermore, we also studied the scalability of the proposed method, which is another key issue determining the feasibility for adopting spectral k-support norm in RPCA task. We generated the data with the sizes of

---

**Figure 5.1:** Corruption ratio versus relative error.

**Figure 5.2:** Rank ratio versus relative error.
1000 × 1000 to 3000 × 3000 and set the corruption ratio to 1%, rank ratio to 10%. As shown in Figure 5.3, the proposed method is more efficient than IALM and PSSV. In the experiment, we also used the truncated SVD to approximately solve the SVT operator, i.e. proximal mapping of nuclear norm. Therefore, the proposed method costs comparable computation of per-iteration. Nevertheless, our method chooses optimal step size adaptive to the smoothness of the dual objective, which can explain why it is faster than IALM and PSSV. By contrast, FWT is much faster than all algorithms because it only computes the top singular value and corresponding vector that can be much more efficient than truncated SVD. As a result, our method, by avoiding full SVD if otherwise ADM is applied, makes the spectral k-support norm efficient enough to use compared to prevalent proximal mapping-based nuclear norm regularized methods. Although not as fast as FWT, the proposed method has the better recovery performance.

Figure 5.3: Data dimension versus CPU time.
Table 5.1: Videos used in the experiment

<table>
<thead>
<tr>
<th></th>
<th>campus</th>
<th>lobby</th>
</tr>
</thead>
<tbody>
<tr>
<td>frame size</td>
<td>160 × 128</td>
<td>160 × 128</td>
</tr>
<tr>
<td># of frames</td>
<td>1,439</td>
<td>1,536</td>
</tr>
<tr>
<td>M size</td>
<td>20,480 × 4,317</td>
<td>20,480 × 4,608</td>
</tr>
</tbody>
</table>

Figure 5.4: Background modeling results on campus and lobby dataset.

5.4.2 Background Modeling on Surveillance Videos

In this experiment, we considered modeling background in surveillance videos captured by a fixed camera. When stacking each frame as column vectors to form the input data matrix, the relative static background can be assumed to be low rank, while the foreground (e.g. human, car movements) can be modeled as sparse noise. Table 5.1 summarizes the dataset used in this experiment. Since we dealt with color videos, in which each frame is described by three sub-matrices, we vectorized these matrices from each frame and stack them together to form a large matrix. Therefore, the row size of large input matrix equals length times width of the frame and the column size is three times of the frame number in the video.

Figure 5.4 shows the extraction performance of a sample frame of the campus dataset (left three columns) and lobby dataset (right three columns). In the raw

[http://perception.i2r.a-star.edu.sg/bkmodel/bkindex.html](http://perception.i2r.a-star.edu.sg/bkmodel/bkindex.html)
image, the foreground mainly contains two pedestrians in the middle of the frame, one in dark shirts, while the other is in white, and a car at the left corner of the frame. The background extracted by spectral k-support norm is obviously better than that by (partial) nuclear norm, in which there are still vague contours of the pedestrian in white shirt in the middle and car in the left from the background extracted by (partial) nuclear norm. The rightmost three columns of Figure 5.4 present a pretty challenging sample frame from lobby dataset, where the two men stand in the middle of the frame for a moment leading them hard to be separate from background. In this case, (partial) nuclear norm is unable to remove these two men from the background. The spectral k-support norm is able to completely remove the man on the right and the man on the left only leaves with a vague contour. As a result, the spectral k-support norm has better recovery performance. Also, the running time indicates that our algorithm is efficient.

In conclusion, this experiment indicates that spectral k-support norm is superior than (partial) nuclear norm for recovering low rank matrix under sparse noise. Also, the proposed method is scalable to large scale tasks that makes the spectral k-support norm feasible to be applied for robust low rank subspace learning.

5.4.3 Face Reconstruction

In this experiment, we consider the face reconstruction task, where front face images are taken under varying conditions like changing illumination. When stacking all vectorized face together, the shadow and specularity caused by changing environment can be treated as sparse noise and the underling low rank matrix is the desired face image to be recovered. We used part of the Extended Yale-B Face Database-B (i.e. subjects 1 to 10 of 38 subjects in total), which contains 64 frontal face pictures of 192 × 168 pixels in each subject. When stacking them together, the input matrix is of size 32256 × 640, which is not very large compared to experiment in the previous subsection. In this case, IALM and PSSV are faster than the proposed method. A snapshot of the reconstruction result on sample images is illustrated in Figure 5.5.
Visually, the spectral k-support norm outperforms both nuclear norm and partial sum of nuclear norm.

5.5 Summary

In this chapter, we have studied robust low rank subspace learning problem with spectral k-support norm to promote the low rank property. Our method can utilize both the squared spectral k-support norm and itself. For both formulations, we consider a sparse norm fitting error ball constrained low rank optimization problem and transform it to the dual objective form. Solving the dual problem only involves a linear subproblem called polar operator and a projection onto the unit sparse ball, which allow us to avoid expensive proximal mapping of the spectral k-support norm. Furthermore, by studying the (sub)gradient of the dual norm of the more generalized spectral k-support norm, we have incorporated a line search strategy that is able to adapt to smoothness change. Experiment result on both synthetic and real datasets with background modeling and face reconstruction have successfully demonstrated the superiority of the proposed method in comparison with the existing counterparts.
Chapter 6

Robust Low-Rank Tensor Minimization via a New Tensor Spectral k-Support Norm

6.1 Introduction

Multidimensional data, formally referred to as tensors, are high-order generalizations to vectors (i.e. first-order tensors) and matrices (i.e. second-order tensors). Tensor is a natural form of many real world data that appears in various areas ranging from image and video analysis in computer vision [154, 88, 87, 100], social network analysis and recommendation system [12] in data mining [125, 83], to signal processing [153, 44], bioinformatics [107], and so on. One prominent example in computer vision and image processing is natural color images, which are 3-way tensors of size \(n_1 \times n_2 \times 3\), where each of the three frontal slices corresponds to a color channel. In practice, the collected tensors are often: 1) having exact or approximate intrinsic low-rank structure; 2) missing some entries due to unavailability or instrument failure; 3) contaminated with arbitrary corruption. Robust tensor principal component analysis and robust low-rank tensor completion, jointly referred to as robust low-rank tensor minimization (RLTM) hereafter, are popular tools for robustly recovering such
complex multi-way data.

In the literature, based on different tensor decomposition algebraic frameworks and their accompany rank definitions, there exist three lines for low-rank tensor estimation, i.e. CANDECOMP/PARAFAC (CP) decomposition model \cite{19,71}, Tucker decomposition model \cite{132}, and tensor singular value decomposition (t-SVD) model \cite{73}. The CP model defines the rank to be the smallest number of rank one tensor decomposition, which then approximates a tensor as sum of rank-one outer products. The CP model has difficulty in determining the CP rank (known to be NP-hard problem). Also, its convex relaxation is ill-posed \cite{54,98}. By contrast, the Tucker model is more tractable, which unfolds a tensor to matrices along each mode (i.e. a single dimension) and defines the rank to be the matrix rank of each unfolded matrix. Many methods use sum of matrix nuclear norm (SNN) of each matricization to convexify the Tucker rank \cite{75,87,123,115}. A tensor is then folded back from the low-rank matrices. Albeit more favored than CP model in certain applications, it fails to exploit the correlations between modes and the unfolding and folding processes tend to discard internal structure information of the tensor. In addition, each mode of matricization has the same number of entries with the original tensor, which leads to heavy computational burden for large size tensors.

A more promising approach which has received increasing interests is the recently proposed t-SVD model \cite{73}. The t-SVD framework decomposes a tensor into a SVD-structure similar to the matrix SVD, which is based on a new defined tensor-tensor product (t-Product)\cite{73}. The t-SVD naturally arises a new tensor tubal rank definition. By seeking low-rankness in terms of the tubal rank, t-SVD based methods expect to better capture the intrinsic structure of a tensor without much loss of correlation information as opposed to matricization of the Tucker model. By mimicking the relationship between matrix rank and matrix nuclear norm, most existing work is to use the tensor nuclear norm (TNN) as convex surrogate, which has achieved state-of-the-art performance in various computer vision and image processing tasks. For example, image and video completion (also called inpainting)
In terms of computational cost, TNN is equivalently defined as the sum of the matrix nuclear norm of each frontal slices after Fourier transformation, whose sizes are smaller than matricization along modes. It reduces the computational cost with a certain degree when compared to the Tucker model. Meanwhile, unlike the matrix case where one can simple relaxing the matrix rank, TNN is introduced based on two different tubal rank quantities. One in [88] is by relaxing on the sum of tubal multi-rank, while the other in [154, 153] is on the tubal multi-rank. Subsequently, they result in two different but related TNN formulations.

In additional, as pointed out by [88], the TNN form by [154, 153] is not dual to the tensor spectral norm, which leads to inconsistency with matrix NN properties.

The rational behind the tensor nuclear norm relaxation, similar to the relationship between $\ell_1$ norm and cardinality in the vector case as well as between nuclear norm and rank in the matrix case, is probably that the nuclear norm is the tightest convex relaxation of the rank function on the tensor spectral norm ball. However, relaxing on the spectral norm ball can be less optimal. For example, in the vector cardinality case, papers [88, 30] show that seeking convex surrogate within unit $\ell_2$ norm ball results into superior performance in sparse regression and feature selection tasks. In the matrix rank case, papers [95, 97] show that the convex relaxation of rank function within unit Frobenius norm ball is superior than the nuclear norm. Hence, it is questionable to exclusively relaxing the t-SVD based rank within unit tensor spectral norm ball.

We may ask: Whether it is possible to derive other form convex surrogate to t-SVD ranks? Whether the new norm allows as convenient formulation as in TNN case that can be presented by matrices norms of the frontal slices in the Fourier domain? Furthermore, most TNN based methods resort to alternating direction method of multipliers (ADMM) [18] for optimization, which computes a proximal operator of the nuclear norm (also known as singular value thresholding) in each iteration. The per-iteration complexity is $O(n_1n_2n_3 \log(n_3) + n_1n_2n_3 \min\{n_1, n_2\})$, which is super-linear with respect to the input tensor of size $\mathbb{R}^{n_1 \times n_2 \times n_3}$. Can we
develop alternative optimization procedure for the new convex surrogate with smaller
complexity so that the t-SVD based methods can scale to a larger dataset like
high-resolution color images and videos?

In this chapter, we focus on the new t-SVD framework and consider the important
models of robust tensor completion and principal component analysis as examples.
First, we revisit the derivation of the two different TNN norms by introducing a
general TNN that includes both as a special case. Rather than obtaining TNN by
relaxing on the tubal multi-rank or the average of the tubal multi-rank within a unit
tensor spectral norm ball, we deduce a general TNN by relaxing the sum of the multi-rank within an $\alpha$-scaled tensor spectral norm ball. When $\alpha = 1$, the general TNN takes the form as defined by [154, 153], and when $\alpha = n_3$, it is the same as the one defined by [88]. Second, apart from TNN, we propose a new tensor norm, called tensor spectral $k$-support norm, as an alternative convex relaxation for sum of tubal multi-rank. Rather than relaxing the tubal rank within a tensor spectral norm ball, which exactly gives rise to the tensor nuclear norm, we propose to relax within a scaled tensor Frobenius norm ball. In particular, we derive the closed-form formulation of the new relaxation, in terms of the matrix spectral $k$-support norm of the frontal slices of the tensor in the Fourier domain, through which we observe that our tensor spectral $k$-support norm is a more general and flexible relaxation, which interpolates between tensor nuclear norm and tensor Frobenius norm. Instead of computing the norm independently for each frontal slice in the Fourier domain, the new norm also contains cross-slices factors which acts as a global factor to further capture the interrelationship of the tensor as a whole.

On the optimization side, we develop two procedures for optimizing in both
primal and dual formulation. In the primal form, we also build upon the ADMM
optimization procedure, where we present the proximal operator for the tensor
spectral $k$-support norm. This proximal mapping can also be conveniently obtained
in the Fourier domain and converted to the proximal mapping of the vector $k$-support
norm on the singular values of the tensor. This method shares the same overall
per-iteration complexity of $O(n_1 n_2 n_3 \log(n_3) + n_1 n_2 n_3 \min\{n_1, n_2\})$ as that for TNN.

In order to scale to large data size, we propose to optimize in a greedy way with the dual objective. Instead of directly solving the Lagrangian dual formulation by ADMM-type methods like the existing approaches, we further convert it by taking Fenchel dual. To optimize the equivalent Fenchel dual objective, we then apply the dual accelerated proximal gradient descent algorithm (APG) \cite{102,149}. We show that the dual APG requires to evaluate the polar operator, rather than proximal operator, with respect to the tensor spectral $k$-support norm, which turns out to require only nearly linear computation (i.e. $O(n_1 n_2 n_3 \log(n_3) + kn_1 n_2 n_3)$) with respect to input tensor size. The dual algorithm is greedy in nature because we construct the low-rank tensor by a convex combination of the intermediate variable of the atom tensor, which is based on the polar operator of TSP-$k$ norm. In particular, TNN can be viewed as combination of sum of tubal rank 1 atomic tensors. The dual optimization procedure implies that sum of tubal rank $k$ atomic tensors allowing $k$ other than 1 can be more general and better choice.

In summary, our contributions in this chapter are three-fold:

1. We propose a new tensor spectral $k$-support norm for tensor tubal rank relaxation, which interpolates between tensor nuclear norm and tensor Frobenius norm, showing great flexibility and generality;

2. We provide both ADMM-based algorithm for medium size data in the primal formulation as well as a greedy optimization for large size data in the dual formulation.

3. We conduct an extensive empirical study of the new norm and the scalable algorithm with both synthetic and real image and video datasets with both medium and large size data.

Furthermore, potential advantages of our TSP-$k$ norm over TNN include:

1. The tensor Frobenius norm factor of TSP-$k$ norm contains additional global information, which can be helpful for better capturing the intrinsic structure
among the entire tensor.

2. Rather than imposing sparsity penalties with $\ell_1$ norm on all singular values, TSP-$k$ only sums $\ell_1$ norm over the minor singular values, which can avoid over penalizing large singular values that tends to leading to skewed estimation.

3. Through the dual optimization procedure, inducing low-rankness with TSP-$k$ results in decomposing the tensor into linear combinations of sum of tubal multi-rank $k$ atom tensors. TNN exclusively decomposes with $k = 1$, which can lead to inferior estimation performance, as real tensors can have various intrinsic decomposition with $k > 1$. TSP-$k$ provides such flexible choice of $k$.

In addition, this chapter extends our preliminary work, which focuses on matrix case [23] only, in the following three aspects:

1. While the previous work focused on matrices, this chapter generalizes to third-order tensors, where the definition and derivation of the tensor spectral $k$-support norm is very different from the matrix one.

2. Along the derivation of the optimization algorithms, new computation components are developed for the tensor case. That is, new proximal operator and new polar operator of the tensor spectral $k$-support norm is established.

3. Experiments are conducted on tasks with natural tensor representations.

The rest of this chapter is organized as follows. In Section II, we describe additional notation and background of the tensor t-SVD algebraic framework. In Section III, we introduce the new tensor spectral $k$-support norm and discuss its properties. In Section IV, we present an optimization algorithm for solving the objective function in the primal form based on ADMM algorithm. In Section V, we present the scalable algorithm, including the Fenchel dual conversion, algorithm design, computational complexity and convergence analysis. Experimental results are reported in Section VI. Finally, we draw a conclusion in Section VII.
6.2 Additional Notation

In this section, after introducing the notation, we review the recently proposed t-SVD algebraic framework, its associated tensor tubal rank and two existing tensor nuclear norms.

For a vector $v$, we use the Matlab command $[v^\dagger, \text{idx}] = \text{sort}(v, \text{‘descend’})$ to represent the operation that sorts $v$ in a decreasing order with the sorted vector stored in $v^\dagger$ and the indices in $\text{idx}$ (i.e. $v(\text{idx})$ is equal to $v^\dagger$).

6.3 A New Convex Relaxation for Robust Tensor Recovery: Tensor Spectral $k$-Support Norm

In this section, we first propose a new tensor norm under the t-SVD framework through a different convex relaxation. The new tensor norm contains TNN and tensor Frobenius norm as special cases. We then introduce two robust low-rank tensor recovery tasks with the new norm as examples.

6.3.1 The New Tensor Spectral $k$-Support Norm

We motivate our new tensor norm by revisiting the relationship between the two existing different TNN notion in [88, 118, 154, 153] and the sum of tubal rank, which is not used by any of the existing methods in convex relaxation. In [88], the authors observe that the averaged TNN is the tightest convex relaxation of the average of the tubal multi-rank within the unit spectral norm ball, while the authors of [118, 154, 153] obtain the summed TNN by relaxing the tubal multi-rank. Instead, the following Proposition 6.3.1 unifies both exiting TNN dentitions by showing that it can also be viewed as the tightest convex relaxation of the sum of the tubal multi-rank within the scaled tensor spectral norm ball. The proof can be found in the Appendix.

**Proposition 6.3.1.** Consider $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and the set

$$\mathcal{C}_k^{(sp)} = \{\mathcal{A} : \text{rank}_{sum}(\mathcal{A}) \leq k, \|\mathcal{A}\|_2 \leq \alpha\}. \quad (6.3.1)$$
Then, the convex hull \( \text{conv}(\cdot) \) of \( C_{k}^{(sp)} \) is given by

\[
\text{conv}(C_{k}^{(sp)}) = \{A : \|A\|_{t^*,\alpha} \leq k, \|A\|_{2} \leq \alpha \}. \tag{6.3.2}
\]

That is, the general TNN of \( \|A\|_{t^*,\alpha} \) is the convex envelop of the sum of the tubal multi-rank within the \( \alpha \)-scaled tensor spectral norm ball. In particular,

- the averaged TNN is a special case of general TNN with \( \alpha = n_3 \);
- the summed TNN is a special case of general TNN with \( \alpha = 1 \).

Instead of relaxing the sum of tubal multi-rank within the scaled tensor spectral norm ball \( C_{k}^{(sp)} \) to obtain TNN, this chapter proposes a new tensor norm by seeking an alternative convex relaxation within a scaled tensor Frobenius norm ball:

\[
C_{k}^{(Fro)} = \{A : \text{rank}_{\text{sum}}(A) \leq k, \|A\|_{F} \leq \sqrt{n_3} \}. \tag{6.3.3}
\]

In particular, with \( \alpha = \sqrt{n_3} \), the following Definition defines the new tensor norm.

**Definition 6.3.2.** The Tensor Spectral \( k \)-Support norm (TSP-\( k \) norm) \( \| \cdot \|_{\text{tsp,k}} \) is defined to be the norm whose unit ball is the convex hull of set \( C_{k}^{(Fro)} \) (i.e. \( \text{conv}(C_{k}^{(Fro)}) \)).

In a similar form like Eq. (6.3.2), the TSP-\( k \) norm satisfies

\[
\text{conv}(C_{k}^{(Fro)}) = \{A : \|A\|_{\text{tsp,k}} \leq k, \|A\|_{F} \leq \sqrt{n_3} \}. \tag{6.3.4}
\]

That is, the TSP-\( k \) norm is the convex envelop of the sum of the tubal rank within the \( \sqrt{n_3} \)-scaled tensor Frobenius norm ball. As in TNN case, TSP-\( k \) norm can also be efficiently dealt with by taking FFT. The next Proposition details the relation between TSP-\( k \) norm of \( A \) with the vector \( k \)-support norm (denoted by \( \| \cdot \|_{v^p,k} \)) of the singular values of \( \widehat{A} \) and the matrix spectral \( k \)-support norm (denoted by \( \| \cdot \|_{msp,k} \)) of the block diagonal matrix \( \widehat{A} \). Proof can be found in the Appendix.

**Proposition 6.3.3.** For tensor \( A \), let \( \widehat{A}, \tilde{A} \) and \( \sigma_{\tilde{A}} \) be FFT-transformed tensor, block diagonal matrix and the singular values of \( \widehat{A} \), correspondingly. The TSP-\( k \)
norm has the following relationships with the $k$-support norm of $\sigma_{\hat{A}}$ and the spectral $k$-support norm of $\hat{A}$ as

$$\|A\|_{tsp,k} = \frac{1}{n_3} \|\sigma_{\hat{A}}\|_{vp,k} = \frac{1}{n_3} \|\hat{A}\|_{msp,k}.$$  \hfill (6.3.5)

The dual norm of the TSP-$k$ norm, denoted by $\| \cdot \|_{tsp,k}^*$, is detailed in the following Proposition.

**Proposition 6.3.4.** Under the same notation as in Proposition 6.3.3, the dual TSP-$k$ norm of $A$ can be computed as,

$$\|A\|_{tsp,k}^* = \|\sigma(\hat{A})[1:k]\|_2 = \|\sigma(\hat{A})\|_{vp,k}^* = \|\hat{A}\|_{msp,k}^*.$$ \hfill (6.3.6)

Sort $\sigma_{\hat{A}}$ in nonincreasing order and denote the result vector by $\sigma_{\hat{A}}^\downarrow$. According to Proposition 6.3.3 and the computation of the vector $k$-support norm [3], TSP-$k$ norm has the following explicit computation as

$$\|A\|_{tsp,k} = \frac{1}{n_3} \left[ \sum_{j=1}^{k-l-1} (\sigma_{\hat{A}}^\downarrow_j)^2 + \frac{1}{l+1} \left( \sum_{j=k-l}^{D} (\sigma_{\hat{A}}^\downarrow_j)^2 \right) \right]^\frac{1}{2},$$ \hfill (6.3.7)

where $l$ satisfies $(\sigma_{\hat{A}}^\downarrow_{k-l-1}) > \frac{1}{l+1} \sum_{j=k-l}^{D} (\sigma_{\hat{A}}^\downarrow_j)^2 > (\sigma_{\hat{A}}^\downarrow_{k-l})$. By Eq. (6.3.7), the index $l$ divides $\sigma_{\hat{A}}^\downarrow$ into larger part $(\sigma_{\hat{A}}^\downarrow_L) = (\sigma_{\hat{A}}^\downarrow)[1:k-l-1]$ and smaller part $(\sigma_{\hat{A}}^\downarrow_S) = (\sigma_{\hat{A}}^\downarrow)_{k-l:D}$. The TSP-$k$ is a combination of the $\ell_2$-norm of the larger part and the $\ell_1$-norm of the smaller part, i.e.

$$\|A\|_{tsp,k} = \frac{1}{n_3} \left( \| (\sigma_{\hat{A}}^\downarrow_L) \|_2^2 + \frac{1}{l+1} \sum_{j=1}^{D} (\sigma_{\hat{A}}^\downarrow_S)^2 \right)^\frac{1}{2}.$$ \hfill (6.3.8)

Hence, the TSP-$k$ norm contains both the tensor nuclear norm factor and the tensor Frobenius factors. Formally, the following Proposition shows that the TSP-$k$ norm interpolates between the tensor nuclear norm and the tensor Frobenius norm. Proof can be found in the Appendix.

**Proposition 6.3.5.** The tensor spectral $k$-support norm becomes tensor nuclear norm when $k = 1$, while becomes tensor Frobenius norm when $k = D$.

By the preceding Proposition, both TNN and tensor Frobenius norm are special cases of TSP-$k$ norm.
6.3.2 Robust low-rank Tensor Recovery with TSP-$k$ Norm

In this chapter, we apply the TSP-$k$ norm to two important robust low-rank tensor minimization tasks. The first is robust tensor principal component analysis \cite{88,64}, which aims to recover a low-rank tensor $L \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and a sparse tensor $E \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ from corrupted observation $X \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. With TSP-$k$ norm as low-rank inducing regularizer, it takes the form as in Example 1.

Example 1. (Robust Tensor PCA)
\[
\min_{L,E} \frac{1}{2} \|L\|_{tsp,k}^2 + \lambda \|E\|_1, \quad s.t. \quad X = L + E,
\]
where $\lambda > 0$ is a regularization parameter.

There are several other variants to the robust tensor PCA model. For example, \cite{160} substitute $\|S\|_1$ with $\|S\|_{2,1}$ to capture outliers and \cite{56} incorporates an additional total variation regularizer for a moving object detection task. Such variations are either on the sparsity-inducing part or introduces additional regularization, which are irrelevant to the low-rank inducing part. Since our focus is on the low-rank inducing part, we do not consider these variants in this chapter.

The second example is robust low-rank tensor completion, which is more challenging as the observed tensor is simultaneously subjected to missing entries and random corruption. Although considered under low-rank matrix setting \cite{22,100}, it is yet known whether its tensor analogue via t-SVD framework also has superior performance. With TSP-$k$ norm, the task is formalized in Example 2.

Example 2. (Robust low-rank Tensor Completion)
\[
\min_{L,E} \frac{1}{2} \|L\|_{tsp,k}^2 + \lambda \|E\|_1, \quad s.t. \quad \Psi_\Omega(X - L) = \Psi_\Omega(E),
\]
where $\Omega$ denotes the observed tensor entries such that $\Psi_\Omega(A)_{ijk} = A_{ijk}$ for $(i,j,k) \in \Omega$ and $\Psi_\Omega(A)_{ijk} = 0$ for $(i,j,k) \in \Omega^\perp$ (we let $\Psi_\Omega(A)_{ijk} = 0$). $\lambda$ is a regularization parameter.
Furthermore, we take a general approach to the above two tasks, by considering the robust low-rank tensor minimization problem as:

$$\min_{\mathcal{L}, \mathcal{E}} \frac{1}{2} \| \mathcal{L} \|_{tsp,k}^2 + \lambda \| \mathcal{E} \|_1, \quad s.t. \quad \mathcal{M}(\mathcal{X} - \mathcal{L}) = \mathcal{E},$$

(6.3.11)

where \( \mathcal{M} \) can be as general as a linear tensor operator defined as \( \mathcal{M}(\mathcal{A}) = \mathcal{M} \ast \mathcal{A} \) (\( \ast \) is the t-Product). Under this general modeling, Example 1 and 2 are special cases with \( \mathcal{M} \) being the identity mapping and the element-wise projection \( \mathcal{P}_\Omega(\cdot) \), correspondingly. The next two sections presents two optimization algorithms for the general model in Eq. (6.3.11).

6.4 An ADMM-based Optimization Algorithm for Medium-size Data

6.4.1 Proximal Operator for TSP-\( k \) Norm-based Regularizer

The core building block for the ADMM based method is the proximal operator for the TSP-\( k \) norm. We show that similar to TNN case, the proximal operator for the TSP-\( k \) norm also has convenient computation by taking FFT and converting the tensor problem to the singular vectors of each frontal slices in the Fourier domain.

The proximal operator solves a second-order optimization subproblem, which is formally defined as follows.

**Definition 6.4.1. (Proximal Operator [109])** The proximal operator of a closed proper convex function \( r(x) \) at \( t \) is defined as

$$\text{Prox}_r(t) = \arg\min_{x} r(x) + \frac{1}{2} \| x - v \|_2^2.$$  

(6.4.12)

For TSP-\( k \) norm-based regularizer in Eq.(6.3.11), the proximal operator has the following formulation,

$$\text{Prox}_{\frac{1}{2\beta} \| \cdot \|_{tsp,k}}^\mathcal{L}(\mathcal{T}) = \arg\min_{\mathcal{L}} \left[ \frac{1}{2} \| \mathcal{L} - \mathcal{T} \|_F^2 + \frac{1}{2\beta} \| \mathcal{L} \|_{tsp,k}^2 \right].$$  

(6.4.13)
where \( \beta \) is a step-size related constant. The formulation of the dual TSP-\( k \) norm is much simpler than TSP-\( k \) norm itself, by comparing Eq.(6.3.6) with Eq.(6.3.7).

The proximal operator of function \( r \) and its Fenchel conjugate \( r^* \) has the following relationship \([102]\),

\[
t = \text{Prox}_r(t) + \text{Prox}_{r^*}(t). \tag{6.4.14}
\]

It is more convenient to calculate the proximal operator of the primal \( \frac{1}{2} \| \cdot \|_{tsp,k}^2 \) by instead computing the proximal operator of the dual \( (\frac{1}{2} \| \cdot \|_{tsp,k}^*)^2 \) and convert back according to

\[
\mathcal{J} = \text{Prox}_{\frac{1}{2\beta} \| \cdot \|_{tsp,k}^2} (\mathcal{J}) + \text{Prox}_{\frac{2}{\beta}(\| \cdot \|_{tsp,k}^*)^2} (\mathcal{J}). \tag{6.4.15}
\]

The next Proposition describes the exact computation of the proximal operator of the dual \( (\frac{1}{2} \| \cdot \|_{tsp,k}^*)^2 \), which is obtained by converting the subproblem optimization from the tensor variable to the singular values in the Fourier domain and then applying the proximal operator of the dual \( k \)-support norm to the vector of singular values.

**Proposition 6.4.2.** Let \( \hat{T} \) be the FFT-transformed tensor of \( \mathcal{J} \) and denote all singular values of \( \hat{T} \) by \( \sigma_{\hat{T}} = [\sigma_{\hat{T}}^{(1)}, \ldots, \sigma_{\hat{T}}^{(i)}, \ldots, \sigma_{\hat{T}}^{(n_3)}] \in \mathbb{R}^D \), where \( \sigma_{\hat{T}}^{(i)} \in \mathbb{R}^{\min\{n_1,n_2\}} \) are the singular values of the \( i \)-th frontal slice \( \hat{T}^{(i)} \). Let \( L^\# = \text{Prox}_{\frac{2}{\beta}(\| \cdot \|_{tsp,k}^*)^2} (\mathcal{J}) \). Let \( [\sigma_{\hat{T}}^\dagger, \text{idx}] = \text{sort}(\sigma_{\hat{T}}, \text{\textquotesingle descend\textquotesingle}) \). Then, the corresponding \( \sigma_{L^\#}^\dagger \) is

\[
(\sigma_{\hat{T}}^\dagger(\text{idx}))_j = \begin{cases} 
\frac{n_3}{\beta} (\sigma_{\hat{T}}^\dagger)_j, & j < k_{\text{low}} \\
\frac{n_3}{\beta} \sum_{j \leq k_{\text{low}}} (\sigma_{\hat{T}}^\dagger)_j - \frac{n_3}{\beta} \frac{k_{\text{upp}}}{(k_{\text{low}}+1) + \frac{n_3}{\beta} (k_{\text{upp}}-k)}, & j \in \mathbb{I}_k^* \\
(\sigma_{\hat{T}}^\dagger)_j, & j > k_{\text{upp}},
\end{cases} \tag{6.4.16}
\]

where \( \mathbb{I}_k^* = [k_{\text{low}}, k_{\text{upp}}] \) is the maximum subset containing \( k \) such that the following conditions hold

\[
\begin{align*}
(\sigma_{\hat{T}}^\dagger)_k^\text{low} &< \frac{\frac{n_3}{\beta}}{1+\frac{n_3}{\beta}} \sum_{j \leq k_{\text{low}}} (\sigma_{\hat{T}}^\dagger)_j - \frac{n_3}{\beta} \frac{k_{\text{upp}}}{(k_{\text{low}}+1) + \frac{n_3}{\beta} (k_{\text{upp}}-k)}, \\
(\sigma_{\hat{T}}^\dagger)_k^\text{upp} &> \frac{\frac{n_3}{\beta}}{1+\frac{n_3}{\beta}} \frac{k_{\text{upp}}}{(k_{\text{low}}+1) + \frac{n_3}{\beta} (k_{\text{upp}}-k)}. \tag{6.4.17}
\end{align*}
\]

The maximality of \( \mathbb{I}_k^* \) means that 1) including one more a larger \( (\sigma_{\hat{T}}^\dagger)_j \) (i.e. \( (\sigma_{\hat{T}}^\dagger)_{k_{\text{low}}-1} \)) then the first inequality of Eq.(6.4.17) cannot hold; 2) including one more
**Algorithm 14** Proximal Mapping for TSP-$k$ Norm: $\text{Prox}_{\frac{1}{2\beta} \| \cdot \|_{\text{TSP},k}}^{}(\mathbf{J})$

**Input:** $\mathbf{J}$ and $\beta$

1: $\hat{\mathbf{J}} = \text{fft}(\mathbf{J}, [\cdot], 3)$;

2: for $i = 1, ..., n_3$ do

3: $[\hat{\mathbf{U}}^{(i)}, \text{diag}(\sigma_{\hat{\mathbf{J}}}^{(i)}), \hat{\mathbf{V}}^{(i)}] = \text{svd}(\hat{\mathbf{J}}^{(i)})$;

4: end for

5: $[\sigma_{\hat{\mathbf{J}}}^{\downarrow}, \text{idx}] = \text{sort}(\sigma_{\hat{\mathbf{J}}}, \text{‘descend’})$;

6: $k^{\text{low}} = \text{RepeatBinarySearch}(1, k, \sigma_{\hat{\mathbf{J}}}^{\downarrow})$;

7: $k^{\text{upp}} = \text{RepeatBinarySearch}(k, D, \sigma_{\hat{\mathbf{J}}}^{\downarrow})$;

8: Compute $\sigma_{\hat{\mathbf{J}}}^{\downarrow}$ according to Eq. (6.4.16);

9: $\sigma_{\hat{\mathbf{L}}}^{\#}(\text{idx}) = \sigma_{\hat{\mathbf{L}}}^{\downarrow}$;

10: for $i = 1, ..., n_3$ do

11: $(\hat{\mathbf{L}}^{\#})^{(i)} = \hat{\mathbf{U}}^{(i)} \text{diag}(\sigma_{\hat{\mathbf{L}}}^{(i)}) (\hat{\mathbf{V}}^{(i)})^\top$;

12: end for

13: $\mathbf{L}^{\#} = \text{ifft}(\hat{\mathbf{L}}^{\#}, [\cdot], 3)$;

**Output:** $\text{Prox}_{\frac{1}{2\beta} \| \cdot \|_{\text{TSP},k}}^{}(\mathbf{J}) = \mathbf{J} - \mathbf{L}^{\#}$

smaller ($\sigma_{\hat{\mathbf{J}}}^{\downarrow}$) (i.e. ($\sigma_{\hat{\mathbf{J}}}^{\downarrow}$)$_{k^{\text{upp}}+1}$) then the second inequality of Eq. (6.4.17) cannot hold. $\mathbf{L}^{\#}$ can be obtained by two repeated binary search over $[1, k]$ and $[k, D]$ given the searching conditions Eq. (6.4.17). Based on Proposition 6.4.2 and Eq. (6.4.15), the proximal operator of $\frac{1}{2} \| \cdot \|_{\text{TSP},k}^2$ is summarized in the following Corollary.

**Corollary 6.4.3.** The proximal operator of $\frac{1}{2} \| \cdot \|_{\text{TSP},k}^2$ at $\mathbf{J}$ with constant $\beta$ can be computed by

$$\text{Prox}_{\frac{1}{2\beta} \| \cdot \|_{\text{TSP},k}}^{}(\mathbf{J}) = \mathbf{J} - \mathbf{L}^{\#},$$

(6.4.18)

where $\mathbf{L}^{\#}$ is obtained by Proposition 6.4.2.

In summary, Algorithm 14 describes the procedure for the proximal mapping of the TSP-$k$ norm.
6.4.2 Preconditioned ADMM-based Optimization Algorithm

The ADMM based method mainly takes alternative updating for the $L$ and $E$ variables. In the following, $\rho$ is a penalty parameter from the augmented Lagrangian formulation and $\eta$ is a constant from linearization of the map $\mathcal{M}$. We provide the update for $L$ and $E$ in the chapter, while the detailed derivation can be found in the Appendix. Algorithm 15 summarizes the procedure.

1. Update of $L$:

$$L_t = \text{Prox}_{\frac{1}{2\rho\eta}||^{2}_{2,\rho,k}} \left( L_{t-1} + \frac{J_{t-1}}{\rho\eta} + \frac{1}{\eta} \mathcal{M}^\top (\mathcal{M} (X - L_{t-1}) - E_{t-1}) \right).$$

(6.4.19)

The computation of the proximal map in Eq. (6.4.19) has been given by Proposition 6.4.2 and detailed in Algorithm 14. In particular, to apply the general update to Examples 1 and 2 it suffices to set $\eta = 1$ and substitute $\mathcal{M}$ by identity map and $\mathcal{P}_0$, correspondingly.

2. Update of $E$:

$$E_t = \text{Prox}_{\frac{1}{\rho}||_{1}} \left( \mathcal{M} (X - L_t) + \frac{J_{t-1}}{\rho} \right),$$

(6.4.20)

where the proximal operator of the $\ell_1$ norm is well-known and can be efficiently computed by the element-wise soft-thresholding operation, i.e. with $\mathcal{J} = (\mathcal{M} (X - L_t) + \frac{1}{\rho} J_{t-1})$,

$$(\text{Prox}_{\frac{1}{\rho}||_{1}} (\mathcal{J}))_{ijk} = \text{sign}(\mathcal{J}_{ijk}) \max\{|\mathcal{J}_{ijk}| - \frac{\lambda}{\rho}, 0\}. \quad (6.4.21)$$

6.4.3 Computational Complexity

A step-by-step complexity analysis for Algorithm 15, including the subroutine Algorithm 14, is provided in the Appendix. To summarize, the per-iteration computational complexity is $O(n_1 n_2 n_3 \min\{n_1, n_2\})$, which is dominated by step 2-4 and step 10-12 for computing the proximal map of TSP-$k$ norm. The high super-linear complexity of both steps is attributed to the full SVD of the $n_3$ frontal slices in the Fourier domain, which are indispensable for computing the proximal map since every singular value element is involved (see step 6-8 of Algorithm 14).
Algorithm 15 Preconditioned ADMM for (6.3.11)

**Input:** Initialize $J_0, L_0, E_0$ and set parameters $T, \rho$.

1: for $t = 0, 1, \ldots, T - 1$ do

2: $L_t = \text{Prox}_{\frac{1}{2\rho} \| \cdot \|_{\text{tsp}, k}^2} \left( L_{t-1} + \frac{1}{\eta} \nabla^T (M(X - L_{t-1}) - E_{t-1}) + \frac{J_{t-1}}{\rho} \right)$ by Algorithm 14;

3: $E_t = \text{Prox}_{\frac{1}{\rho} \| \cdot \|_1} \left( M(X - L_t) + \frac{J_{t-1}}{\rho} \right)$ by Eq. (6.4.21);

4: $J_t = J_{t-1} + \rho (M(X - L_t) - E_t)$;

5: end for

**Output:** $L_T, E_T$

### 6.5 A Greedy Dual Optimization Algorithm for Larger-Size Data

The more convenient computation of the dual norm in computing the proximal map of TSP-$k$ norm inspire us to convert the problem to the dual form at the first place, rather than to resort to dual form only at proximal mapping at each iteration. In this section, we first show an equivalent objective by taking Fenchel conjugate. Then, we show the dual objective can be solved by a universal primal dual method \[14\], where the core building block involves a linear subproblem for TSP-$k$ norm, called polar operator. We provide the computation for the polar operator, which costs only linear per-iteration complexity. This method is in essence greedy by adding one atom computed according to the polar operator at one iteration. It not only provides a scalable optimization, but also shows that the low-rank tensor can be viewed as a linear combination of sum of tubal multi-rank $k$-tensors. From this perspective, TNN is a special case that builds a low-rank tensor exclusively by $k = 1$ combinations, which can be suboptimal for some applications where the intrinsic low rank tensor is a $k > 1$ combination of atoms.
6.5.1 Equivalent Dual Reformulation

Instead of considering the regularized form as in Eq.(6.5.22) we instead consider the constrained form:

\[
\text{argmin}_{\mathcal{L}, \mathcal{E}} \frac{1}{2} \|\mathcal{L}\|^2_{tsp,k}, \text{ s.t. } \|\mathcal{E}\|_1 \leq \tau \text{ and } \mathcal{M}(\mathcal{X} - \mathcal{L}) = \mathcal{E},
\]

which is equivalent to the regularized form in Eq.(6.5.22) with proper pair of \(\tau\) and \(\lambda\). The constraints amounts to \(\|\mathcal{M}(\mathcal{X} - \mathcal{L})\|_1 \leq \tau\). By directly signifying the tolerance on the misfit, it is considered more natural than regularization formulation [2]. Also, for some applications where the misfit can be estimated, the constrained form Eq.(6.5.22) can better utilize it as a priori. The next Proposition converts Eq.(6.5.22) to a dual TSP-\(k\) norm related equivalence by Fenchel conjugation. Proof can be found in the Appendix.

**Proposition 6.5.1.** Let \(\mathcal{J}\) denote the dual variable. The primal formulation in Eq.(6.5.22) has the following equivalent dual form,

\[
\min_{\mathcal{J}} \mathcal{D}(\mathcal{J}) = \min_{\mathcal{J}} \mathcal{f}(\mathcal{J}) + \mathcal{h}(\mathcal{J}),
\]

where \(\mathcal{f}(\mathcal{J}) = \frac{1}{2} (\| - \mathcal{M}^\top(\mathcal{J})\|_{tsp,k}^*)^2 + \langle \mathcal{J}, \mathcal{M}(\mathcal{X}) \rangle\),

\[
\text{and } \mathcal{h}(\mathcal{J}) = \tau \| - \mathcal{J}\|_\infty.
\]

In the following, we call \(\mathcal{f}\) as the dual loss function and \(\mathcal{h}\) as the dual regularizer. To solve the dual objective with gradient descent based methods, the next Proposition reveals a particular choice of the (sub)gradient of \(\mathcal{f}(\mathcal{J})\). Proof can be found in the Appendix.

**Proposition 6.5.2.** The (sub)gradient of the dual loss function \(\mathcal{f}\) at \(\mathcal{J}\), denoted by \(\mathcal{g}(\mathcal{J})\), can be computed as

\[
\mathcal{g}(\mathcal{J}) = -\mathcal{M}(\mathcal{L}^\#) + \mathcal{M}(\mathcal{X}),
\]

where \(\mathcal{L}^\# = (\| - \mathcal{M}^\top(\mathcal{J})\|_{tsp,k}^*) \cdot \mathcal{A}^\#\),

\[
\text{and } \mathcal{A}^\# = \arg\max_{\|\mathcal{A}\|_{tsp,k} \leq 1} \langle -\mathcal{M}^\top(\mathcal{J}), \mathcal{A} \rangle.
\]
The core part for computing $g(J)$ is to compute $A^\#$ by solving a linear subproblem in Eq.(6.5.28), which is called the polar operator of TSP-$k$ norm. By intuition, the linear subproblem should be more efficient to compute than the second subproblem for solving the proximal map. The next subsection derives the closed-form solution for the polar operator.

### 6.5.2 Polar Operator for TSP-$k$ Norm

**Definition 6.5.3. (Polar Operator [114, 145])** For a norm $r(x)$, the polar operator at $t$ is defined as

$$\text{Polar Operator}_r(v) = \arg \max_x \{ \langle x, v \rangle : r(x) \leq 1 \}. \quad (6.5.29)$$

We show that the polar operator of TSP-$k$ norm can be computed with complexity nearly linear with respect to the input tensor size, which is more favored than super-linear complexity dependence of the proximal mapping. The following is called the polar operator of TSP-$k$ norm on $J$:

$$A^\# = \arg \max_A \{ \langle J, A \rangle : \| A \|_{lsp,k} \leq 1 \}, \quad (6.5.30)$$

which can be computed according to the following Proposition. Proof can be found in the Appendix. Algorithm 16 summarizes the computation.

**Proposition 6.5.4.** Let $\tilde{J} = \text{fft}(J, [], 3)$, $\tilde{J}^{(i)} = \tilde{U}^{(i)} \text{diag}(\sigma^{(i)}_{\tilde{J}})(\tilde{V}^{(i)})^\top$, $\sigma_{\tilde{J}} = [\sigma^{(1)}_{\tilde{J}}, ..., \sigma^{(n_3)}_{\tilde{J}}]$ and $[\sigma^{\perp}_{\tilde{J}, \text{idx}}] = \text{sort}(\sigma_{\tilde{J}}, \text{‘descend’})$. The polar operator result of Eq.(6.5.30) satisfies,

$$A^\# = \text{ifft}(\tilde{A}^\#, [], 3), \quad (\tilde{A}^\#)^{(i)} = \tilde{U}^{(i)} \text{diag}(\sigma^{(i)}_{\tilde{J}})(\tilde{V}^{(i)})^\top,$$

where $\sigma_{\tilde{J}}^\perp(\text{idx}) = \sigma_{\tilde{J}}^{\perp}$ and

$$(\sigma^{\perp}_{\tilde{A}^\#})_j = \begin{cases} \frac{n_3(\sigma^{\perp}_{\tilde{J}})_j}{\| \sigma^{\perp}_{\tilde{J}} \|_2}, & j \in [1 : k], \\ 0, & j \in [k : D]. \end{cases} \quad (6.5.32)$$
Finally, we have the computation for \( \mathcal{L}^\# \) in Eq.\( \text{(6.5.28)} \), referred as the atom of TSP-\( k \) hereafter in brief, based on the polar operator computation, which is summarized in the following Corollary \( \text{6.5.5} \). Proof is in the Appendix.

**Corollary 6.5.5.** Let \( \mathcal{J} = -\mathcal{M}^\top(\mathcal{J}) \) and \( \hat{\mathcal{J}}, \sigma_{\hat{\mathcal{J}}}, \sigma_{\hat{\mathcal{J}}}^{-\frac{1}{2}}, \text{idx}, \hat{\mathcal{U}}^{(i)} \) and \( \hat{\mathcal{V}}^{(i)} \) be the same notation as defined in Proposition \( \text{6.5.4} \). Then \( \mathcal{L}^\# = \text{ifft}(\hat{\mathcal{L}}^\#, [, 3) \), where \( \forall i \in [n_3] \),

\[
(\hat{\mathcal{L}}^#)^{(i)} = \hat{\mathcal{U}}^{(i)} \text{diag}(\sigma_{\hat{\mathcal{L}}^#}^{(i)})(\hat{\mathcal{V}}^{(i)})^\top, \quad \text{and} \quad (6.5.33)
\]

\[
(\sigma_{\hat{\mathcal{L}}^#}(\text{idx})) = \begin{cases} n_3(\sigma_{\hat{\mathcal{L}}^#}^{-\frac{1}{2}})_j, & j \in [1 : k], \\ 0, & j \in [k : D]. \end{cases} \quad (6.5.34)
\]

Eq.\( \text{(6.5.32)} \) indicates that \( \hat{\mathcal{A}}^# \) only depends on the largest \( k \) singular values among \( \sigma^{(1)}, ... , \sigma^{(n_3)} \). Hence, it suffices to calculate the leading \( k \) singular values of each \( \hat{\mathcal{X}}^{(i)} \), which contain \( \sigma_j^# \) for \( j \in [1, k] \) for sure. That is, to compute \( \hat{\mathcal{A}}^# \), we only need to calculate partial svd, by Lanczos method [82] or Power method [46], which cost only \( O(kn_1n_2n_3) \) per-iteration computation and is much smaller than \( O(n_1n_2n_3 \min(n_1, n_2)) \) of the proximal mapping in ADMM method, since in practice \( k \) is much smaller than \( \min(n_1, n_2) \).

### 6.5.3 Universal Primal-Dual Optimization Based Algorithm

Equipped with the dual formulation, its (sub)gradient and the polar operator for TSP-\( k \) norm in the preceding subsections, we then develop a scalable greedy dual algorithm based on the universal primal-dual optimization [149]. The algorithm is summarized in Algorithm [17] where each iteration is composed by two ingredients:

1. **Update of \( \mathcal{J} \):** An accelerated proximal gradient descent (APG) is used for updating the Lagrangian dual variable \( \mathcal{J}_t \):

   \[
   \mathcal{J}_{t+1} = \arg \min_{\mathcal{J}} \mathcal{f}(\mathcal{J}) + \langle \mathcal{g}_t, \mathcal{J} - \mathcal{J}_t \rangle + \frac{H_t}{2} \| \mathcal{J} - \mathcal{J}_t \|_F^2 + \mathcal{h}(\mathcal{J}); \quad (6.5.35)
   \]

   \[
   \mathcal{J}_{t+1} = \mathcal{J}_{t+1} + \frac{\theta_t - 1}{\theta_{t+1}}(\mathcal{J}_{t+1} - \mathcal{J}_t), \quad (6.5.36)
   \]

13
Algorithm 16 Computing Polar Operator for TSP-$k$ Norm in Eq.(6.5.30): $\mathcal{A}^\# = \text{Polar}\text{Operator}(\mathcal{J})$

**Input:** $\mathcal{J}$

1: $\tilde{\mathcal{J}} = \text{fft}(\mathcal{J}, [[], 3]);$
2: for $i = 1, \ldots, n_3$ do
3:   $[\tilde{U}^{(i)}, \text{diag}(\sigma_{\tilde{\mathcal{J}}^{(i)}}, \tilde{V}^{(i)})] = \text{partial}\_\text{svd}(\tilde{\mathcal{J}}^{(i)}, k);$
4: end for
5: $[\sigma_{\tilde{\mathcal{J}}}^{\perp}, \text{idx}] = \text{sort}(\sigma_{\tilde{\mathcal{J}}}, \text{‘descend’}) \sigma_{\tilde{\mathcal{J}}} = [\sigma_{\tilde{\mathcal{J}}}^{(1)}, \ldots, \sigma_{\tilde{\mathcal{J}}}^{(n_3)}];$
6: Compute $\sigma_{\mathcal{A}^\#}^{\perp}$ by Eq.(6.5.32);
7: Rearrange $\sigma_{\mathcal{A}^\#}^{\perp}$ back to corresponding frontal slices by $\sigma_{\mathcal{A}^\#}^{\perp}(\text{idx}) = \sigma_{\mathcal{A}^\#}^{\perp};$
8: for $i = 1, \ldots, n_3$ do
9:   $(\mathcal{A}^\#)^{(i)} = \tilde{U}^{(i)} \text{diag}((\sigma_{\mathcal{A}^\#}^{\perp})^{(i)})(\tilde{V}^{(i)})^\top;$
10: end for
11: $\mathcal{A}^\# = \text{ifft}(\tilde{\mathcal{A}}^\#, [[], 3]);$

**Output:** Polar\_Operator($\mathcal{J}$) = $\mathcal{A}^\#$

where $\theta_{t+1}$ is a constant sequence updated according to $\theta_{t+1} = \frac{1+\sqrt{1+4\theta_t^2}}{2}$ with initialization $\theta_1 = 1$, and $\tilde{\mathcal{J}}$ is an acceleration sequence kept by the APG method. $H_t$ can be a constant parameter such that the R.H.S. of Eq.(6.5.35) is an upper estimation of the dual objective, which is also the reciprocal of the step size. A better choice of $H_t$ that enables the algorithm to adapt to both the degree $\nu$ and magnitude of the Hölder smoothness notion of the dual loss function is provided in the Appendix. Eq.(6.5.35) is the proximal mapping of the dual regularizer of $\ell_1$ norm, i.e. $h(\mathcal{J}) = \tau \|\mathcal{J}\|_\infty$, as

$$\mathcal{J}_{t+1} = \text{Prox}_{\frac{\tau^2}{H_t}} \|\|_\infty (\mathcal{J}_t - \frac{1}{H_t} \mathcal{g}(\mathcal{J}_t)). \quad (6.5.37)$$

The proximal mapping of the dual norm can be performed by the projection on the unit ball of the original primal norm, as

$$\mathcal{J}_{t+1} = \left(\mathcal{J}_t - \frac{1}{H_t} \mathcal{g}(\mathcal{J}_t)\right) - \frac{\tau}{H_t} \text{Proj}_\|\|_1 \left(\frac{1}{\tau/H_t} (\mathcal{J}_t - \frac{1}{H_t} \mathcal{g}(\mathcal{J}_t))\right). \quad (6.5.38)$$
2. Update of $\mathcal{L}$: Alongside the dual variable updating, a linear combination step is used for updating the primal variable $\mathcal{L}_t$:

$$\mathcal{L}_{t+1} = (1 - \gamma^t)\mathcal{L}_t + \gamma^t\mathcal{L}_t^\#,$$  

(6.5.39)

where the weighting sequence is $\gamma^t = \frac{\theta_t/H_t}{\sum_{j=1}^{k} \theta_j/H_j}$. The update of Eq. (6.5.39) is by nature greedy as it combines one atom to the low rank estimation at one time. Thus, we can observe from Eq. (6.5.39) that the low rank tensor induced by TSP-$k$ norm is a linear combination of sum of tubal multi-rank $k$ tensors. By Proposition 6.3.5, TNN is limited to $k = 1$ combination of atoms, which can be suboptimal for modeling tensors with intrinsic $k > 1$ combination of atoms.

The update of Eq. (6.5.39) is also closely related to the conditional gradient (or called Frank-Wolfe method). A key difference between Eq. (6.5.39) and the conditional gradient update is that the weight $\gamma^t$ takes into consideration the smoothness $H_t$ of the dual loss function when the linear-search subroutine in Algorithm 1 of the Appendix is adopted.

### 6.5.4 Complexity and Convergence Analysis

A step-by-step computational analysis can be found in the Appendix. One dominating computational cost comes from the polar operator step. Unlike ADMM methods for TNN and TSP-$k$, which takes $O(n_1n_2n_3 \min(n_1, n_2))$ for proximal mapping, ours costs only $O(kn_1n_2n_3)$. Note that for some real problems, where the underlying tensor is of very low-rank, $k$ here can be a very small constant, rendering our complexity a much more scalable for large size tensor problems. Considering another dominant computational costs of $\text{fft}$ and $\text{ifft}$ steps which are shared by all tensor SVD methods, our greedy dual method effectively reduces the complexity down from the super-linear complexity to nearly linear of $O((k + \log n_3)n_1n_2n_3)$.

The iteration complexity is based on the universal primal-dual algorithm, as summarized in the next Proposition.

**Proposition 6.5.6.** For the primal variable $\mathcal{L}_T$ to achieve $\epsilon$ error, Algorithm takes worst case number of iterations by $O(\inf_{\nu \in [0,1]}(\frac{H_\nu}{\epsilon})^{\frac{1}{1 + \nu}}).$
Algorithm 17 Scalable Tensor Spectral $k$-Support Norm Regularized Robust Low-Rank Tensor Minimization

**Input**: $X, \mathcal{M}, \tau, H_0, T$;

1: for $t = 0, 1, ..., T - 1$ do
2: Compute $\mathcal{L}_t^\#$ by Eq.(6.5.33) by calling $\mathcal{A}_t^\# = \text{Polar Operator}(\mathcal{J}_t)$;
3: Compute $g(\mathcal{J}_t)$ by Eq.(6.5.26);
4: Backtracking Line Search for $H_t$ (Optional);
5: Update dual variable: $\mathcal{J}_{t+1} = \text{Prox}_{\frac{\tau}{H_t}}(\mathcal{J}_t - \frac{1}{H_t}g(\mathcal{J}_t))$ according to Eq.(6.5.38);
6: Compute the primal recovery weight: $\gamma_t = \frac{\theta_t/H_t}{\sum_{j=1}^p \theta_j/H_j}$;
7: Update the acceleration sequence: $\theta_{t+1} = \frac{1+\sqrt{1+4\tau_t^2}}{2}$;
8: Update the interpolation sequence: $\mathcal{J}_{t+1} = \mathcal{J}_{t+1} + \frac{\theta_t-1}{\theta_{t+1}}(\mathcal{J}_{t+1} - \mathcal{J}_t)$;
9: Update the primal variable: $\mathcal{L}_{t+1} = (1 - \gamma_t)\mathcal{L}_t + \gamma_t\mathcal{L}_t^\#$;
10: end for

**Output**: $\mathcal{L}_T$

In above, $\nu$ is the degree of the Hölder smoothness. For example, for smooth objective, $\nu = 1$ and the worst case iteration number is of order $O\left(\frac{1}{\epsilon^{1/2}}\right)$, which is as fast as APG for regularized smooth primal loss problems.

### 6.6 Experiment

#### 6.6.1 Experiments on medium size datasets

**Synthetic Dataset**

We first compare the proposed TSP-$k$ norm with TNN norm on synthetic dataset. To generate the low-rank ground truth tensor $X_{gt}$, we uniformly sample from $[0, 1]$ an $n \times n \times n$ tensor and then truncated it to $\text{Rank}_{\text{tubal}}$ by t-SVD. We then add arbitrary corruption by randomly sampling $\rho$ of the $n^3$ entries and set them to $-20$ or $20$ with equal probability. For fairness of comparison, we use the same ADMM optimization algorithm for both methods. The balancing parameter $\lambda$ is set to $1/\sqrt{\max(n_1, n_2)n_3}$.
Figure 6.1: Comparison of PSNR value for image denoising experiment on all images of Berkeley Segmentation Database.
Table 6.1: Recovery results on $n \times n \times n$ random data with different tubal rank.

<table>
<thead>
<tr>
<th></th>
<th>$\ell_1$ relative error</th>
<th>$\ell_2$ relative error</th>
<th></th>
<th>$\ell_1$ relative error</th>
<th>$\ell_2$ relative error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TNN</td>
<td>TSP-$k$</td>
<td>TNN</td>
<td>TSP-$k$</td>
<td></td>
</tr>
<tr>
<td>$n$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>$2.70 \times 10^{-6}$</td>
<td>$7.19 \times 10^{-7}$</td>
<td>$3.50 \times 10^{-6}$</td>
<td>$9.73 \times 10^{-7}$</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>$3.69 \times 10^{-7}$</td>
<td>$2.56 \times 10^{-7}$</td>
<td>$4.63 \times 10^{-7}$</td>
<td>$3.47 \times 10^{-7}$</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>$1.19 \times 10^{-6}$</td>
<td>$1.59 \times 10^{-7}$</td>
<td>$1.50 \times 10^{-6}$</td>
<td>$2.17 \times 10^{-7}$</td>
<td></td>
</tr>
</tbody>
</table>

$Rank_{tubal}(X_{gt}) = 20$

<table>
<thead>
<tr>
<th></th>
<th>$\ell_1$ relative error</th>
<th>$\ell_2$ relative error</th>
<th></th>
<th>$\ell_1$ relative error</th>
<th>$\ell_2$ relative error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TNN</td>
<td>TSP-$k$</td>
<td>TNN</td>
<td>TSP-$k$</td>
<td></td>
</tr>
<tr>
<td>$n$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>$1.44 \times 10^{-6}$</td>
<td>$7.84 \times 10^{-7}$</td>
<td>$2.03 \times 10^{-6}$</td>
<td>$1.09 \times 10^{-6}$</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>$3.69 \times 10^{-7}$</td>
<td>$2.56 \times 10^{-7}$</td>
<td>$4.63 \times 10^{-7}$</td>
<td>$3.47 \times 10^{-7}$</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>$7.60 \times 10^{-7}$</td>
<td>$3.24 \times 10^{-7}$</td>
<td>$9.71 \times 10^{-7}$</td>
<td>$4.29 \times 10^{-7}$</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.2: Example images and recovery results. The indices of selected images from top to bottom are 006,020,087,144,193, where 020,087,144 and 193 are popularly demonstrated by previous related work. From left to right: original image, corrupted image; Recovered images by RPCA, PSSV, SNN, TNN, TSP-$k$. 

118
Table 6.2: Comparison of PSNR values on example images from BSD.

<table>
<thead>
<tr>
<th>Image index</th>
<th>RPCA</th>
<th>PSSV</th>
<th>SNN</th>
<th>TNN</th>
<th>TSP-k</th>
</tr>
</thead>
<tbody>
<tr>
<td>006</td>
<td>31.16</td>
<td>33.02</td>
<td>33.62</td>
<td>35.67</td>
<td><strong>39.88</strong></td>
</tr>
<tr>
<td>020</td>
<td>25.18</td>
<td>27.43</td>
<td>27.77</td>
<td>28.84</td>
<td><strong>32.52</strong></td>
</tr>
<tr>
<td>087</td>
<td>28.91</td>
<td>30.17</td>
<td>30.66</td>
<td>32.58</td>
<td><strong>37.25</strong></td>
</tr>
<tr>
<td>144</td>
<td>24.31</td>
<td>26.07</td>
<td>26.51</td>
<td>29.27</td>
<td><strong>32.91</strong></td>
</tr>
<tr>
<td>193</td>
<td>22.39</td>
<td>24.66</td>
<td>24.24</td>
<td>26.43</td>
<td><strong>30.56</strong></td>
</tr>
<tr>
<td>All 200 images (avg)</td>
<td>25.53</td>
<td>27.92</td>
<td>27.63</td>
<td>29.46</td>
<td><strong>33.04</strong></td>
</tr>
</tbody>
</table>

Figure 6.3: Comparison of PSNR value of each frame for simultaneous video completion and denoising experiment.

for TNN (we use the form in Definition 2.4.11) as suggested in [88]. We empirically set \( k = \text{Rank}_{tubal} + 2 \) for TSP-\( k \) and tune \( \lambda \) by grid search.

Table 6.1 reports the recovery results with different \( n \) in terms of both \( \ell_1 \) and \( \ell_2 \) relative error (relative to the ground truth). As can be seen, TSP-\( k \) achieves smaller error with varying tensor sizes.

**Image Denoising**

We consider the image denoising task with the entire 200 images from the Berkeley Segmentation Database (BSD) [94]. The BSD dataset contains 200 color images of medium size (e.g. \( 321 \times 481 \times 3 \)) and the content spans a wide variety of natural scenes and objects. Please note that nature color images are often considered to be approximately low-rank, since their leading singular values of a small number dominate the main information and a large number of singular values are very close
Figure 6.4: Sample frames of simultaneous video completion and denoising experiment on videos: Bike, Basketball and Walking (from top to bottom). From left to right: original frame, missing and corrupted frame, recovered frame by TNN, recovered frame by TSP-$k$.

to zero. We generate random corruptions by randomly sampling 10% of the 3-way tensor entries and set them to random values in $[0, 255]$, which results up to 30% of the pixels to be randomly corrupted. We compare our method with tensor low-rank inducing norms TNN and SNN. We also compare with matrix low-rank inducing norm based methods RPCA (matrix NN) and PSSV [106], the latter of which is a nonconvex relaxation of the nuclear norm by discarding top singular values out of the nuclear norm. For fair of comparison, we again use ADMM-type methods for optimizing all tensor and matrix-based norms. We set $tol = 10^{-5}$ for tensor based methods (we also test $10^{-7}$, but the results barely change) and $tol = 10^{-7}$ for matrix based methods, which are default values chosen by the corresponding authors. As for $\lambda$, we follow [88] for the compared methods. For our TSP-$k$ norm, we find that with the parameter pair $[k, \lambda] = [5, 20]$, TSP-$k$ is superior than all compared low-rank inducing norms on 140 images of the entire 200 images. While for the remaining 60 cases, TSP-$k$ achieves the best performance by slightly adjusting the choice to pairs like $[3, 20]$ (for 43 images) and $[7, 15]$ (for 13 images) and the remaining three also use $k = 5$ but set $\lambda$ to 10 or 15. In terms of parameter setting, we can see that: 1)
the $\lambda$ parameter is not difficult to tune to get superior performance, since TSP-$k$ norm performance much better than all compared norms on 183 of 200 images (over 90\%) with a single $\lambda = 20$; 2) the setting of $k = 3, 5, 7$ indicates that natural images have different underlying linear combination of rank $k$ sum of tubal multi-rank and $k = 1$ (choice of TNN) is not optimal.

The recovery performance is summarized in Fig. 6.1, which reports the PSNR values of all norms on all 200 images in BSD. Fig.6.1 exhibits recovered images on example images. Table 6.2 shows the PSNR values of the example images and the average PSNR value on the entire BSD dataset. From both the visual and quantitative experiment results, we summarize that 1) the t-SVD methods (TNN and TSP-$k$) are better than matrix-based norms (RPCA and PSSV) and Tucker model based norm (SNN); 2) TSP-$k$ norm is a better convex relaxation than TNN for the sum of tubal multi-rank; 3) to achieve the superior performance, the choice of parameters for TSP-$k$ is not difficult.

**Video recovery from observation with simultaneous corrupted and missing entries**

We consider the video recovery task, where the video frames are corrupted by random noise and some values in the RGB channels are missing. We consider three video subsequences with different characteristics: 1) the Bike clip where the camera is moving to chase the moving bike, so that the background is gradually changing and a single object in the foreground is moving; 2) the Basketball clip where the camera chasing the players, so that the background is gradually changing and multiple objects in the foreground is moving; 3) the Walking clip where a steady surveillance camera captures a walking human, so that the background is still and a main object in the foreground is moving. All three videos are of medium sizes with frame dimensions $640 \times 360$, $576 \times 432$ and $384 \times 288$. We randomly selects 30 consecutive frames for the experiment. With each frame being one frontal slice of the tensor, we reshape the video data into tensors of corresponding sizes $640 \times 1080 \times 30$, $576 \times 1296 \times 30$ and $303 \times 121$. 
Figure 6.5: Large size image recovery with 20% missing and 20% corruption. From top to bottom are images: bird, architecture, flower, human and tower. From left to right are: original image, input corrupted image, FWT output at 2000s CPU time, TNN and TSP-\(k\) at 1000s CPU time, TNN and TSP-\(k\) at 1500s CPU time, TNN and TSP-\(k\) at 2000s CPU time.

384 \times 864 \times 30, by vertically concatenating the RGB channel of each color frame into a frontal slice. We add 10\% of all tensor values in [0, 255] randomly as the corruption and set 20\% tensor values as random missing. Fig. 6.3 presents the PSNR values of every frame in all video clips. Our method achieves better recovery results on all frames of the three video clips. Fig. 6.4 shows the samples frames and our method is also better by visual comparison.

6.6.2 Experiments on Larger Size Datasets

In this part, we consider the large size observation tensor setting as in Section V, where the existing proximal ADMM based t-SVD methods have scalability problem. Please note that there is no existing work has ever studied the t-SVD based low-rank tensor method under this larger scale robust low-rank tensor completion setting. We show that our TSP-\(k\) norm with the proposed greedy dual method in Algorithm 17 is able to get better recovery result on large size color images and high resolution
Figure 6.6: Comparison of Large size image recovery with 20% missing and 20% corruption, which include $\ell_1$ (top) and $\ell_2$ (bottom) relative error versus CPU time. From left to right are results on images: Bird, Architecture, Flower, Human and Tower.

videos in much less CPU time.

**Large size image recovery from observation with simultaneous missing entries and random corruption**

We utilize large size color images collected from Flickr as shown in the first column of Fig. 6.5, which have different content including: bird, ancient architecture, flower, human and Eiffel Tower. All images have frame sizes exceeding $3000 \times 3000$, which is much larger than the images in BSD. The second column of 6.5 shows the observation images, which are generated by setting 20% of the corresponding original image tensor entries with random corruption in $[0, 255]$ and randomly selecting 20% entries as missing. We compare with the ADMM-based TNN method as well as matrix NN method solved by a mixed Frank-Wolfe method called FWT in [100] which is also greedy in nature. As shown in the third column of Fig. 6.5, the matrix NN has unsatisfiable recovery performance visually even at CPU time 2000s. From the 4th to 9th column of Fig. 6.5 we present the recovery output of TNN and our TSP-$k$ method at CPU time 1000s, 1500s and 2000s. Our method has better recovery result by using same CPU time. In addition, visually, our output at 1000s CPU time is better than the output of TNN at 2000s CPU time, which indicates our method can have better performance by using less than half of the time. Fig. 6.6 shows
the quantitative comparison by plotting $\ell_1$ and $\ell_2$ error versus the CPU time, from which we can see that our method decreases the misfit much faster.

High resolution video recovery from observation with simultaneous missing entries and random corruption

We consider video recovery from random corruption and missing entries on two 1440p resolution video clips with randomly selected 10 consecutive frames, which are of frame size $2560 \times 1440$ and of reshaped tensor size $2560 \times 4320 \times 10$. It is much larger than the videos considered in the preceding section and presents great scalability issue for proximal ADMM based methods due to the full SVD per-iteration. The first is an airport video shot by hand held camera, where the whole image is slightly shaken and the plane in the center of the frame is pushing back. The second is a still camera capturing a bridge, where the water and people on the bridge are moving. Fig. 6.7 presents the sample frames of the original and corrupted and missing entries observation video. Fig. 6.9 shows the output sample frame recovered by ADMM-based TNN and our greedy dual based TSP-$k$ at 2000s, 3000s, 4000s and 5000s CPU time. Visually, our method has much better recovery performance at every reported time. Also, our output at 2000s is better than the output at 5000s returned by TNN. Fig. 6.8 by plotting $\ell_1$ and $\ell_2$ relative error against the ground-truth original video, also confirms that our method recovers low-rank video more efficiently and more accurately.
Figure 6.8: Comparison of relative error of the normalized video tensor against CPU time for simultaneous large scale video completion and denoising experiment. From left to right are $\ell_1$ and $\ell_2$ relative error decrease on the Airport, $\ell_1$ and $\ell_2$ relative error decrease on the Bridge.

Figure 6.9: Comparison of recovered frames. Rows from top to bottoms are: recovered frame of airport video by TNN, recovered frame of airport video by TSP-$k$ with greedy dual method; recovered frame of bridge video by TNN, recovered frame of bridge video by TSP-$k$ with greedy dual method. From left to right: frame recovered at CPU time 2000s, 3000s, 4000s, 5000s.
6.7 Summary

In this chapter, we have studied the robust low-rank tensor minimization problem under the t-SVD framework. We have derived an alternative relaxation to the sum of tubal multi-rank by providing a novel tensor spectral $k$-support norm. In particular, we have shown that TSP-$k$ norm interpolates between TNN and tensor Frobenius norm, which is helpful for preserving more global information of the intrinsic low-rank tensor. We provide two optimization methods for dealing with both medium and larger scale data, based on the primal and dual form of the objective function, correspondingly. Experiments on synthetic, image and video datasets with medium and larger-scale dimensions verified the superiority of TSP-$k$ over TNN for low rank tensor modeling as well as the effectiveness of the two proposed optimization methods for their targeted data scales.
Chapter 7

Uplink Communication Efficient Differentially Private Sparse Optimization with Feature-wise Distributed Data

7.1 Introduction

Empirical risk minimization (ERM) is a fundamental tool for learning useful models from data that are collected from individuals. To avoid breaching the privacy of the individuals, privacy protection mechanism have been considered to ensure the adversary cannot infer any individual data from the output of the learning process. Beginning with the seminal work \[21\], which considers private ERM training under the formal statistical differential privacy notion \[35\], various differentially private optimization algorithms have been developed for training the model with centralized datasets \[124, 72, 5, 128, 62, 68\] and sample-wise distributed datasets \[57, 104, 17\].

Deviating from the above mentioned centralized and sample-wise distributed settings, we consider the feature-wise distributed dataset setting. Such setting appears in many real applications, where the information describing an individual is
collected and held by different parties which can be different sets of sensory systems or different organizations. For example, a person’s medical records are sensitive personal information that can be held by several clinics. Although privacy issue has been considered for these vertically-partitioned datasets \cite{148, 143, 93}, it has rarely been studied with the more restrict differential privacy notion. It would be ideal to make use of all attributes kept by different parties in a distributed fashion, while still ensuring differential privacy.

During distributed training, information sourced from user nodes to the server node, referred to as uplink communication, are computed based on sensitive individual information. On the other hand, the information broadcasted by the server back to users (referred as downlink communication), by the post-processing property of the differential privacy \cite{35}, will maintain differential privacy even without any privacy protection mechanism as long as the uplink communication was private. As such, designing privacy mechanism to prevent adversary from inferring sensitive individual information by spying on the uplink communication is key to private training for distributed dataset. Intuitively, minimizing the uplink communication complexity means less exposure of sensitive data and reduced potential of personal data leakage. Thus, less uplink communication would in general require less privacy protection budget for ensuring differential privacy.

To be specific, we consider the constrained ERM model in this chapter as \[
\min_{\mathbf{x} \in \mathcal{M}} \sum_{i=1}^{n} f(\mathbf{x}; \mathbb{D}_i),
\]
where \( \mathbb{D}_i \) represents the \( i \)-th sample of the \( n \) by \( d \) dataset \( \mathbb{D} \), where \( n \) is the sample size and \( d \) is the feature dimension. With the feature-wise distributed setting, the \( n \times d \) data matrix is partitioned vertically with each disjoint partition held by one of the \( K \) user nodes. \( f(\mathbf{x}) \) is a smooth convex loss function. We assume the convex compact constraint set \( \mathcal{M} \) is coordinate separable as \( \mathcal{M}_1 \times \ldots \times \mathcal{M}_d \). In particular, we elaborate the LASSO problem, where \( f(\mathbf{x}) \) is the least square loss and \( \mathcal{M} \) is the \( \ell_1 \)-norm ball. For this problem, \cite{129} proposes a centralized private Frank-Wolfe algorithm (FW) \cite{60} (as most existing conditional gradient methods are referred by Frank-Wolfe algorithm in the coordinate descent
variants, we will use the term Frank-Wolfe algorithm in this chapter) by adapting the Report-Noisy-Max mechanism \cite{35} for ensuring differential privacy. In particular, they prove that the algorithm has nearly optimal utility guarantee for the private LASSO task, meaning no private algorithm has better utility up to a \log(n) factor.

With the above in mind, we propose a distributed private FW algorithm for solving ERM in an uplink communication efficient way, hoping to obtain the same nearly optimal utility as in the centralized setting. At first glance, it seems straightforward by integrating the Report-Noisy-Max for privacy protection \cite{129} with the existing distributed FW design. However, such direct combinations either 1) does not have known differentially private strategy for communicating active features which is indispensable for computing local partial gradient under feature-wise distributed setting \cite{8}; or 2) requiring each user node to have full replication of the entire feature set, which is undesired as local features need to be communicated with an extra preprocessing step \cite{134}. The seemingly different aspects of incapabilities actually attribute to the same reason: existing randomized block-coordinate FW (BCFW) algorithms have limited convergence guarantees that are only applicable under two simple block sampling distributions. In this regard, this chapter makes primarily two contributions:

1. **BCFW under arbitrary sampling:** we develop a much general convergence analysis for BCFW under arbitrary sampling (BCFW-AS). It enjoys greater flexibility than existing analysis \cite{77,134} whose analysis is highly dependent on the specific samplings. Further, in contrast to existing Expected Separable Overapproximation (ESO) inequality based coordinate descent under arbitrary sampling algorithms \cite{113,112,110,111}, we develop our convergence analysis by introducing a new notion called expected curvature, which is a more fundamental quantity for FW algorithms than ESO-based counterparts.

2. **Private FW for feature-wise distributed dataset:** We provide an uplink communication efficient private algorithm based on the BCFW-AS. By adopting our general convergence guarantee, it has guaranteed utility which preserves the same
nearly optimality for the LASSO task as that in the centralized setting [129]. In addition, compared to the existing work [53] under the same setting, our method has improved overall uplink communication complexity $O(n^{\frac{3}{2}})$ than their $O(n)$.

In the next section, after introducing the notation, we provide a thorough discussion of known convergence guarantees for BCFW algorithms, as well as existing private methods for training under feature-wise distributed setting.

### 7.2 Additional Notation and Preliminary

We use $[d]$ to denote the set $\{1, 2, ..., d\}$ and use bold characters $\mathbf{A}, \mathbf{x}$ for matrices and vectors. $\mathbf{A}^\top$ denotes the transpose of matrix $\mathbf{A}$. The operator $\circ$ denotes the Hadamard multiplication, i.e. element-wise multiplication of vectors or matrices of the same sizes. The superscript is associated with iteration number, e.g. $\mathbf{x}^t$ denotes the decision variable at iteration $t$, while subscripts is associated with indices of the coordinates or different parties. For a random subset $\mathcal{T}$ of $[d]$, $|\mathcal{T}|$ denotes the cardinality of set $\mathcal{P}$. $\mathbf{x}(\mathcal{T}) \in \mathcal{M}_\mathcal{T}$ denotes the vector of length $|\mathcal{T}|$ which only keeps the values of $\mathbf{x}$ indicated by $\mathcal{T}$ and $\mathbf{x}[\mathcal{T}]$ denotes the zero padded one. We use $\nabla f(\mathbf{x})$ to denote the gradient of the loss function $f(\mathbf{x})$ at $\mathbf{x}$ and $\nabla(\mathcal{T}) f(\mathbf{x})$ is the partial gradient taken with respect to subset of coordinates indexed by $\mathcal{T}$. $\| \cdot \|_{(M_i)}$ denotes a primal norm defined on $M_i$ and $\| \cdot \|_{(M_i)^*}$ denotes the associated dual norm. We use $\mathbf{e}_i$ to denote the standard basis vector.

#### 7.2.1 Sampling Distributions and Known Convergence Results for Block Coordinate Frank-Wolfe Algorithms

We describe related Frank-Wolfe algorithms by interpreting them as randomized block coordinate Frank-Wolfe under the corresponding sampling distributions to highlight the association of the convergence analysis with the types of distributions. Table 1 summarizes the previous and our new BCFW algorithms, samplings and convergence guarantees. We follow the naming convention used in existing
arbitrarily sampled coordinate descent papers \cite{113,110,111} for referring several common samplings.

**Elementary sampling:** \cite{60} is the conventional deterministic FW algorithm using the full gradient per-iteration, which can be seen as sampling the coordinates under elementary sampling with set \([d]\), i.e. sampling set \([d]\) with probability one. With the step size \(\gamma^t = \frac{2}{t+2}\), it guarantees \(h^t = f(x^t) - f(x^*) \leq \frac{2C_f}{t+2}\), where \(x^* \in \mathcal{M}\) denotes an optimum and \(C_f\) is the curvature of \(f(x)\) on the whole constraint set \(\mathcal{M}\), which measures the non-linearity of \(f(x)\) on the entire constraint set \(\mathcal{M}\), reflecting the geometric property of \(f(x)\) on \(\mathcal{M}\):

\[
C_f := \sup_{x, s \in \mathcal{M}, \gamma \in [0,1]} \frac{2}{\gamma^2} \left( f(x + \gamma(s - x)) - f(x) - \gamma<s - x, \nabla f(x)> \right).
\]  \hfill (7.2.1)

The algorithm dFW \cite{8} is a distributed FW method. During one communication pass, each worker evaluates the partial linear oracle based on the local features and then sends both the partial linear oracle index and the associated local duality gap value to the sever node for comparison. Subsequently, the partial linear oracle with the maximum local duality gap is selected and sends back to all workers for the next update. However, the updates of the local partial gradient requires sharing of “active features” at each communication round. It is unknown how to communicate active features in a private and communication efficient way.

**Uniform serial sampling:** \cite{77} is a randomized block coordinate Frank-Wolfe (BCFW) method selecting the block to be updated in each iteration according to the uniform serial sampling, i.e. samples one block at each iteration with uniform probability. For analyzing the convergence, \cite{77} designs the step size \(\gamma^t = \frac{2d}{t+2d}\) and introduces the product curvature to obtain \(h^t \leq \frac{2d(C_f^\circ + h^0)}{t+2d}\) primal gap. The product curvature \(C_f^\circ := \sum_{i=1}^{d} C_f^i\), where \(C_f^i\) is the block-wise partial curvature for measuring the non-linearity on individual \(\mathcal{M}_i\),

\[
C_f^i := \sup_{x, s \in \mathcal{M}, \gamma \in [0,1]} \frac{2}{\gamma^2} \left( f(x + \gamma(s[i] - x[i])) - f(x) \\
- \gamma<s(i) - x(i), \nabla_i f(x)> \right).
\]  \hfill (7.2.2)
\(\tau\)-nice sampling: AP-BCFW is a parallel and distributed BCFW method, provided that all user nodes have the full replication of the entire dataset. During one communication pass, each worker uniformly samples one block from all blocks for updating and the server node summarizes \(\tau\) non-duplicate updates (discard duplicate update, e.g. two worker samples the same node). Under ideal computational facility, analyze the convergence by equalizing one communication pass as one iteration of centralized BCFW selecting blocks for updating according to \(\tau\)-nice sampling, i.e. samples \(\tau\) blocks with uniform probability. It requires yet another set of step size 
\[
\gamma_t = \frac{2\tau r}{\tau^2 t + 2d}
\]
and expected set curvature 
\[
C_f^S := \binom{d}{\tau}^{-1} \sum_{S \subseteq [d], |S| = \tau} C_f(S),
\]
where the set curvature \(C_f(S)\) is
\[
C_f^S := \sup_{\mathbf{x}, \mathbf{s} \in \mathcal{M}, \gamma \in [0, 1], |S| = \tau} \frac{2}{\gamma^2} \left( f(\mathbf{x} + \gamma(\mathbf{s}[S] - \mathbf{x}[S])) - f(\mathbf{x}) \right.
\]
\[
- \gamma \langle \mathbf{s}(S) - \mathbf{x}(S), \nabla_f(S)\mathbf{x}(S) \rangle \right).
\]

AP-BCFW is obviously unsuited to the distributed feature set, since it would require copy-and-paste local features to other nodes before computation, which incurs high communication cost and raises privacy concern.

\((K, \tau)\)-distributed sampling: This is probability the simplest sampling scheme for the distributed optimization tasks with disjointly divided local blocks, where each user nodes uniformly sampling \(\tau\) blocks from their local blocks which collectively constitutes \(K \times \tau\) random block updates from \(K\) workers. However, existing random BCFWs do not have convergence guarantee even for this simplest sampling.

Arbitrary sampling: We consider a general BCFW that has guaranteed convergence under arbitrary sampling with two minimal assumptions: 1) the sampling is independent across iterations, i.e. the sampling distribution at the present iteration independent of the sampling of the last iteration; 2) the sampling is proper that any block has nonzero probability to be sampled.
7.3 Block-Coordinate Frank-Wolfe under Arbitrary Sampling

7.3.1 Algorithm Description

Algorithm 18 presents the BCFW-AS algorithm under arbitrary proper sampling \( S \).

For notational convenience, in this section, \( i \in [d] \) can be either single coordinate or block of coordinates that we do not explicitly differentiate them with additional notation, while in the next section it refers to a single coordinate. Denote the probability for sampling block \( i \) by \( p_i \) and collectively by \( p := \{p_1, \ldots, p_d\} \). Let \( p_{\text{min}} \) denote the smallest entry in \( p \).

In each iteration (e.g. \( t \)), line 2 samples a random set of blocks \( T^t \) from \{1, 2, ..., \( d \)\} according to sampling distribution \( S \). To accommodate the injected noise for privacy protection in the next section, and also for wider applicability of BCFW-AS, line 5 allows the partial linear oracle (LO) \( \hat{s}_i \) to be evaluated approximately with inexactness parameter \( \ell_i \), given inexact partial gradient (PG) \( \nabla_{(i)} f(x) \) with parameter \( g^i \) in line 4. For brevity, \( \hat{s}_i \) is referred as \( \ell_i \)-LO and \( \nabla_{(i)} f(x) \) is referred as \( g^i \)-PG. Convergence analysis shows that as long as the following assumptions hold, BCFW-AS is still guaranteed to converge:

**Assumption 7.3.1.** (Inexact linear oracle and inexact gradient) Let \( \gamma^t \) denote the step size at iteration \( t \), \( C_f^{ES} \) denote expected curvature of function \( f \) with sampling \( S \).

- Let \( \ell_i \) be a constant parameter. The inexact linear oracle \( \hat{s}_i \) satisfies,
  \[
  \langle \hat{s}_i, \nabla_{(i)} f(x) \rangle \leq \min_{s(\iota) \in M_i} \langle s(\iota), \nabla_{(i)} f(x) \rangle + \frac{\ell_i \gamma^t C_f^{ES}}{2}. \tag{7.3.4}
  \]

- Let \( g^i \) be the inexact gradient constant parameter, \( \nabla_{(i)} f(x) \) be the exact partial gradient at \( x \). The partial gradient \( \hat{\nabla}_{(i)} f(x) \) satisfies,
  \[
  \|\hat{\nabla}_{(i)} f(x) - \nabla_{(i)} f(x)\|_{(M_i)} \leq \frac{g^i \gamma^t C_f^{ES}}{2}, \tag{7.3.5}
  \]

where \( \| \cdot \|_{(M_i)} \) is the dual norm of the norm associated with \( M_i \).
Table 7.1: Comparison of BCFW convergence results

<table>
<thead>
<tr>
<th>Sampling</th>
<th>$p_{\text{min}}$</th>
<th>Existing result</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elementary with set $[d]$</td>
<td>1</td>
<td>$\frac{2(1+\delta)C_f}{t+2}$</td>
<td>$\frac{2(h_0+(1+\delta)C_f}{t+2}$</td>
</tr>
<tr>
<td>Uniform Serial</td>
<td>$\frac{1}{d}$</td>
<td>$\frac{2d(h^0+(1+\delta)dC_f^0)}{t+2d}$</td>
<td>$\frac{2d(h^0+(1+\delta)dCE^0_{uni\ seri})}{t+2d}$</td>
</tr>
<tr>
<td>$\tau$-nice</td>
<td>$\frac{\tau}{d}$</td>
<td>$\frac{2d(h^0+(1+\delta)dC_f^\tau)}{\tau^2t+2d}$</td>
<td>$\frac{2d(h^0+(1+\delta)dCE^{\tau}_{nice})}{\tau^2t+2d}$</td>
</tr>
<tr>
<td>$(K, \tau)$-distributed</td>
<td>$\frac{(K\tau)^2}{d^2}$</td>
<td>-</td>
<td>$\frac{2d(K\tau h^0+(1+\delta)dCE^{(K, \tau)}_f)}{(\tau K)^2t+2dK\tau}$</td>
</tr>
<tr>
<td>Arbitrary</td>
<td>$p_{\text{min}}$</td>
<td>-</td>
<td>$\frac{2(h^0+(1+\delta)CE_{S_0})}{p_{\text{min}}t+2}$</td>
</tr>
</tbody>
</table>

In the above assumption, the constant $C_f^{ES}$ is the expected curvature to be introduced in the next subsection. Step 6 then updates the block $i$-th decision variable, for $i \in T^t$. Concisely, we can add up $\hat{s}^t_i$ for all $i \in T^t$ to denote $\hat{s}^t_{[T^t]} = \sum_{i \in T^t} \hat{s}^t_i$. Then, the total update across all blocks being sampled can be summarized into,

$$x_{[T^t]}^{t+1} = x^t + \gamma^t(\hat{s}^t_{[T^t]} - x^t_{[T^t]}) \quad (7.3.6)$$

Algorithm 18 Block-Coordinate Frank-Wolfe Algorithm With Arbitrary Sampling

1: **Input:** Initial feasible variable $x^0$, step sequence $\gamma^t$, sampling distribution $S$, inexactness parameters $\xi_g$ and $\xi_l$, estimation of expected curvature $C_f^{ES}$, maximum iteration $T$;

2: **for** $t = 0, 1, ..., T - 1$ **do**

3: Generate a random set $T^t \subset [d]$, following the distribution $S$;

4: **for each** $i \in T^t$ **do**

5: Compute approximate partial gradient $\nabla_{(i)} f(x^t)$ satisfies eq.(7.3.5);

6: Compute approximate partial linear oracle $\hat{s}^t_{(i)}$ satisfies eq.(7.3.4);

7: Update $x_{(i)}^{t+1} = x_{(i)}^t + \gamma^t(\hat{s}^t_{(i)} - x_{(i)}^t)$;

8: **end for**

9: **end for**

10: **Output:** $x^T$;
7.3.2 Expected Curvature

Before moving to the convergence analysis of BCFW-AS, this subsection introduces a new notion called expected curvature, which will play a key role in the convergence of the BCFW-AS in the next subsection. The expected curvature compactly associates the curvature of the loss function on various directions, which are randomly sampled according to the sampling distribution. Intuitively, instead of measuring the largest deviation of the loss function from its linear approximation along some particular sets of coordinates, such a new quantity should be able to measure the maximum deviation “averaged” over various choices of sets of coordinates selected under the sampling distribution to manifest the interaction of the intrinsic geometric property along different directions and the distribution of the sampling. We formalize the intuition by the following expected curvature definition:

**Definition 7.3.1.** (Expected curvature) The expected curvature of a the loss function $f(x)$ with arbitrary proper sampling $S$ is defined as,

$$C_f^S = \sup_{x, s \in \mathcal{M}} \mathbb{E}_{T \sim S} \left[ \frac{2}{\gamma^2} f(x + \gamma(s_{[T]} - x_{[T]})) - f(x) - \gamma \langle s_{[T]} - x_{[T]}, \nabla f(x) \rangle \right].$$  \hspace{1cm} (7.3.7)

It first calculates the average deviation with various combinations of proper $x, s, \gamma$ under the sampling $S$ and then chooses the largest value calculated from a certain pair of $(x^#, s^#, \gamma^#)$ as the expected curvature. In general, $(x^#, s^#, \gamma^#)$ may not achieve the maximum partial curvature for every possible combination of blocks under the sampling, yet it ensures the overall supremacy after taking into the probability of the appearance of the combinations of blocks. We discuss some properties of the expected curvature by comparing it with: 1) Lipschitz smoothness-based ESO quantities in [113, 112, 111]; 2) Various curvatures used by existing BCFWs under the samplings mentioned on Page 2. The comparisons show that the expected curvature can be much smaller than Lipschitz-based ESO constants and also refine existing curvature constants used in the specific samplings. For the ease of comparison, we make the same assumption as the compared methods given by
Assumption 7.3.2. There is an \( n \) by \( d \) matrix \( A \) such that for all \( x, y \in \mathcal{M} \),
\[
f(y) \leq f(x) + (y - x, \nabla f(x)) + \frac{1}{2}(y - x)\mathbf{A}^\top \mathbf{A}(y - x).
\] (7.3.8)

Comparison with ESO quantity: Similar to conventional curvature quantities that have the Lipschitz smoothness constant times the squared diameter of the constraint set as the upper bound, the expected curvature is upper bounded by the ESO quantity times the squared diameter given by

**Proposition 7.3.2.** Under Assumption \( \text{7.3.2} \), denote the pairwise probability matrix of arbitrary sampling \( S \) by \( \mathbb{P} \), the diameter of \( \mathcal{M}_i \) by \( D_{\mathcal{M}_i} \). By choosing \( \beta = (\beta_1, \ldots, \beta_d) \), where \( \beta_i = \min\{\sigma'(\mathbb{P}), \sigma'(\mathbf{A}^\top \mathbf{A})\}||A_i||_2 \), where \( \sigma'(\mathbb{P}), \sigma'(\mathbf{A}^\top \mathbf{A}) \) are the largest normalized eigenvalues of the matrices \( \mathbb{P} \) and \( \mathbf{A}^\top \mathbf{A} \), then \( C_{\mathcal{E}}^{\mathcal{S}} \leq \sum_{i=1}^d p_i \beta_i D_{\mathcal{M}_i}^2 \).

In the above proposition, \( \beta_i \) is exactly one of the ESO quantity obtained in \( [111] \) under the same assumption. \( [111] \) have also provided many other estimation of \( \beta_i \). We omit those comparisons because we can also show the same result in a similar fashion. The proofs can be found in the Appendix.

Comparison with curvatures of existing BCFW: The next proposition shows the relationship of the expected curvature with those used in the existing BCFW algorithms.

**Proposition 7.3.3.** Recall the global curvature \( C_f \) of \( [60] \) under elementary sampling, the product curvature \( C_f^{\circ} \) of \( [77] \) under uniform serial sampling, and expected set curvature \( C_f^\tau \) of \( [134] \) under \( \tau \)-nice sampling as introduced in Subsection \( \text{7.2.1} \). Then, the following relationships hold,
\[
C_{\mathcal{E}}^{\text{element}} = C_f, \quad C_{\mathcal{E}}^{\text{uni-seri}} \leq \frac{1}{d} C_f^{\circ}, \quad C_{\mathcal{E}}^{\text{nice}} \leq C_f^\tau,
\] (7.3.9)

where all the left hand side terms denote the expected curvature under the corresponding samplings.

By proposition \( \text{7.3.3} \), the expected curvature is always upper bounded by existing specific curvature constants introduced for specific samplings, which will result in refined convergence results as shown in the next subsection.
Exact form of expected curvature under specific samplings: It is possible to calculate an accurate estimation of the expected curvature under certain samplings if the probability is specified, despite its seemingly abstract definition. We illustrate with two examples of \( \tau \)-nice sampling, \((K, \tau)\)-distributed sampling, the latter will be used in the next section to establish the new convergence of the distributed FW algorithm with disjoint blocks and the utility of the private distributed FW algorithm.

**Proposition 7.3.4.** Under Assumption 7.3.2, the expected curvature under \( \tau \)-nice sampling satisfies

\[
C_{\tau, \text{nice}}^E \leq \tau \mu_1 + \tau(\tau - 1)\mu_2, \quad \mu_1 = \sup_{i \in [d]} \|A_i(s_i - x_i)\|_2^2, \\
\mu_2 = \sup_{i,j \in [d], i \neq j} (A_i(s_i - x_j))^\top (A_j(s_i - x_j)).
\]

This matches the expected set curvature calculated in [134], based on which they further provide example curvature quantities for structured SVM and group fused LASSO.

**Proposition 7.3.5.** Under Assumption 7.3.2, the expected curvature under \((K, \tau)\)-distributed sampling satisfies

\[
C_{K, \tau}^E \leq K \tau \mu_1 + K \tau(\tau - 1)\mu_2 + K(K - 1)\tau^2 \mu_3,
\]

where
\[
\mu_1 = \sup_{i \in [d]} \|A_i(s_i - x_i)\|_2^2, \\
\mu_2 = \sup_{i,j \in [d], i \neq j} (A_i(s_i - x_j))^\top (A_j(s_i - x_j)), \\
\mu_3 = \sup_{i \in \mathcal{P}_1,j \in \mathcal{P}_2,k_1 \neq k_2} (A_i(s_i - x_j))^\top (A_j(s_i - x_j)).
\]

We close this subsection by repeating the remarks on curvature in [60] to stress that curvature is a more fundamental quantity for FW algorithm due to 1) it can be much smaller than Lipschitz smoothness based quantity; 2) it is affine invariant and the analysis applies for arbitrary choices of norms. Thus, for the analysis of BCFW-AS, it is necessary to introduce the new expected curvature quantity, because it is more suitable than the existing Lipschitz smoothness related ESO quantity.
7.3.3 Convergence Analysis for BCFW with Arbitrary Sampling

Equipped with the expected curvature, this subsection provides the convergence result of BCFW-AS in Theorem 7.3.6. The proof can be found in the Appendix.

**Theorem 7.3.6.** (Convergence result of BCFW-AS) Let Assumption 1 and 2 hold. Let $D_M$ denote the diameter of constraint set $\mathcal{M}$, $|\mathcal{T}|$ denote the maximum sampling set size among iteration $1, \ldots, t$. Take $\varrho_g = \max_i \varrho_g^i$, $\varrho_l = \max_i \varrho_l^i$, $\varrho = (D_M + |\mathcal{T}|)\varrho_g + \varrho_l|\mathcal{T}|$. For each $t \geq 0$, the iterates $\mathbf{x}^t$ generated by Algorithm 18 with step size $\gamma^t = \frac{2}{p_{min} t + 2}$ satisfy

$$E[f(\mathbf{x}^t)] - f(\mathbf{x}^*) \leq \frac{2(h^0 + \frac{(1+\varrho)C_{ES}^f}{p_{min}})}{p_{min} \cdot t + 2},$$

where $\mathbf{x}^*$ denotes an optimum of the problem, $h^0 := f(x^0) - f(\mathbf{x}^*)$, $p_{min}$ denotes the minimum entry of probability vector $\mathbf{p}$, $C_{ES}^f$ is the expected curvature for arbitrary sampling $\mathcal{S}$.

According to Theorem 7.3.6, instead of designing the step size for various sampling schemes case by case, BCFW-AS introduces a universal choice that only depends on the minimum probability entry $p_{min}$ (note that further line-search for the step size is also possible as considered in [60, 77], but the convergence analysis is still based on the deterministic step size). This universal step-size alleviates design complexity, especially when applied to complex sampling schemes where one may not know full details of the distribution. In particular, in the distributed private LASSO application, participating user nodes may have their own sampling schemes and do not want to reveal, making it impossible to have full knowledge of the overall sampling distribution.

To demonstrate the flexibility of the convergence guarantee provided by Theorem 7.3.6, we recover the convergence results of the previous FW algorithms by simply substituting the corresponding $p_{min}$ and expected curvature obtained in the previous subsection in eq. (7.3.11), which are summarized in Table 17.3.1. In particular, we have
1. **Elementary sampling**: we obtain the same convergence rate and dependence on the curvature;

2. **Uniform serial sampling**: we obtained a slightly refined convergence result compared to [77] since our curvature-related constant is upper bounded by theirs according to Proposition 7.3.3;

3. **τ-nice sampling**: roughly, we recover a compatible result to [134] that with same convergence rate, our dependence on the curvature-related constant is better while the dependence on the initial primal gap is inferior than theirs;

4. **(K, τ)-distributed sampling**: we obtain the convergence for this new sampling by simply substituting $p_{\text{min}} = \frac{(K\tau)^2}{d^2}$ and expected curvature $C_f^{E(K,\tau)}$ in eq. (7.3.10), which shows the generality of Theorem 7.3.6.

Finally, we would like to remark that the convergence developed for arbitrary samplings has the potential to be applied to developing new BCFW algorithms with guaranteed convergence, like a variant with importance sampling scheme.

### 7.4 Uplink Communication Efficient Differentially Private BCFW with Distributed Features

This section considers the private LASSO problem for feature-wise distributed datasets and proposes an uplink communication efficient algorithm based on BCFW-AS algorithm. Two major components are 1) Private index computation subroutine for computing private linear oracle; 2) Private active feature sharing required for computing partial gradient without privacy leakage. The algorithm comes with guaranteed privacy, utility and uplink communication complexity.
7.4.1 Algorithm Description

\( K \) user nodes solve the LASSO problem:

\[
\min_x \frac{1}{2n} \|Ax - y\|_2^2, \quad s.t. \|x\|_1 \leq \eta, \tag{7.4.12}
\]

with the input matrix \( A = [a_1, ..., a_i, ..., a_d] \), where \( a_i \) is a \( n \times 1 \) sparse feature vector with \( nnz(a) \) non-zero entries at most. For simplicity, the features is assumed to be randomly and evenly splitted among user nodes as \( A = [A_1, ..., A_k, ..., A_K] \) and denote the associated coordinate indices as \( \mathcal{P}_k \) for \( A_k \). To adapt the BCFW-AS to LASSO with distributed features, each user node \( k \) randomly samples \( \tau \) coordinate \( T^k_t \) and computes the partial gradient \( \nabla(T^k_t) f(x^t) \) and then evaluates partial linear oracle \( \arg \max_{i \in \mathcal{T}^k_t} \|\nabla_i f(x^t)\|, \quad idx^t_k = \text{sign}(\nabla_i f(x^t)) \cdot i^t_k; \tag{7.4.14} \)

At iteration \( t \), to update the decision variable, the user node only need to upload the signed index \( idx^t_k \) rather than a full partial decision variable. To compute the partial gradient \( \nabla_i f(x^t) \), \( q^t := \frac{1}{n} Ax^t \) cannot be updated by each user independently based solely on local features, which requires additional communication (recall that local gradient can be computed only based on local samples in the sample-wise distributed setting). However, it can be iteratively updated based on the update rule of \( x^t \), as

\[
q^t = (1 - \gamma^{t-1})q^{t-1} + \gamma^{t-1} \sum_{k=1}^{K} -\text{sign}(idx^t_k) \frac{1}{n} a_{idx^t_k}^{t-1}, \tag{4.15}
\]

Hence, in order to update \( q^t \) from \( q^{t-1} \), it suffices to sharing \( K \) local features \( a_{idx^t_k}^{t-1} \). That is, each user node should upload the local feature indicated by \( idx^t_k \), which is referred to active feature. Apparently, at most \( KT \) nonduplicate active features need to be communicated across \( T \) iterations. Compared with [53] which randomly
sketches features at preprocessing which costs $O(n \times r \log(r))$ uplink communication, the active feature communication takes a “share-at-need” strategy which costs only $O(n \times KT)$ communication complexity. Later, we will show this strategy indeed has smaller communication complexity. Two subroutines: private signed index computing and private active feature sharing provides privacy protection to the algorithm. Given privacy budget $(\epsilon, \delta)$, we evenly assign half for each part for simplicity.

**A. Private signed index computation:** The computation of $idx$ can be regarded as selecting from $2|\mathcal{T}_t^k|$ indices whose associated value $\text{sign}(idx)\nabla_{|idx|}f(x)$ reaches maximum. This equivalence suggests the usage of “Report-noisy-max”, a differential privacy building block, to provide privacy protection for the computation of $idx$. In brief, instead of selecting the index based on clean associated value, “Report-noisy-max” selects based on noise-injected associated value. For our problem, it has the updates as follows:

1. For each $i \in \mathcal{T}_k^t$, compute noise injected partial gradient by $v_i = \text{sign}(i)\nabla_i f(x) + \text{pert}$, where pert is a Laplacian noise: $\text{pert} \sim \text{Lap}(\frac{(2G_0/Kn)\sqrt{2T \log(1/(\delta/2))}}{(\pi^2)/2})$.

2. pick $\hat{idx}_k^t = \text{sign}(v_{i^*})i^*$, where $v_{i^*} = \max_{i \in \mathcal{T}_k^t} v_i$.

[129] also uses “Report-noisy-max” in the centralized FW algorithm for linear oracle selection, and our private signed index computation is an adaptation to the distributed case. We emphasize that to develop private FW under feature-wise distributed setting, a more important design is how to share the active features in a private and communication efficient way without deteriorating the optimal utility guarantee, which is presented below.

**B. Private active feature sharing:** We propose to communicate perturbed sketch active features to jointly provide privacy protection as well as reduce communication cost. For $a_{|idx|}^t \in A_{\text{trans}}$, the user node takes $\hat{a}_{|idx|}^t =Ja_{|idx|}^t + \xi$, where $J$ is a $m \times n$ Gaussian sketch matrix [137] (a type of Johnson-Lindenstrauss transformation matrix) and $\xi$ is an $m \times 1$ noise vector with each entry sampled according to $pert \sim \mathcal{N}(0, \pi^2)$, $\pi = \frac{\sigma(J)\sqrt{KT}\sqrt{2\ln(\frac{1}{\epsilon})+\epsilon/2}}{ne/2}$, where $\sigma(J)$ is the leading singular value of $J$. Analysis shows that $m$ can be as small as $m = \Omega(\frac{1}{\epsilon^2} \log \frac{T}{\delta/2})$ with $T = O((ne)^{2/3})$ and $t$ being
Table 7.2: Summarization of Parameters in Theorem 7.4.2

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varrho_g$</td>
<td>$O\left(\frac{(K\text{nnz}(a)+K^{3/2}\sqrt{\text{nnz}(a)m}\sigma(J)^2)/p\text{min}(2\eta+K\gamma)}{nC_f^2\gamma^T}\sqrt{T}\sqrt{(\log(1/\delta)+\epsilon)}\right)$</td>
</tr>
<tr>
<td>$\varrho_t$</td>
<td>$O\left(\frac{2G\eta\sqrt{32T\log(1/(\delta/2))}\log(2\eta K\gamma T)/(K\gamma)}{nC_f^2(K\gamma T)^p}\right)$</td>
</tr>
<tr>
<td>$C_g$</td>
<td>$((K\text{nnz}(a) + K^{3/2}\sqrt{\text{nnz}(a)}\sigma(J)^2)/p\text{min})\sqrt{(\log(1/\delta) + \epsilon)}$</td>
</tr>
<tr>
<td>$C_l$</td>
<td>$2G\eta\sqrt{32\log(1/(\delta/2))}$</td>
</tr>
<tr>
<td>$m$</td>
<td>$\Omega\left(\frac{1}{T^2}\log\frac{T}{\delta/2}\right)$</td>
</tr>
</tbody>
</table>

a small constant parameter (referred to as JL-parameter), which greatly reduces the communication cost for sharing the active features.

We stress that the sketched active feature sharing for reducing communication cost itself is new to distributed FW type algorithms, even without privacy protection design. Also, we provide an analysis showing that with the provided $m$ the algorithm is guaranteed to converge with same $O\left(\frac{1}{T}\right)$ rate. In addition, this sketching is also crucial for ensuring optimal utility guarantee. Note that the gradient inexact parameter is the much smaller $O(\log(n)/n)$ with sketch (see Table 2 and Theorem 7.4.2), compared to that of $O(\sqrt{n}/n)$ without sketch. Hence, the sketched features require much less noise injection. With inexactness parameter as large as $O(\sqrt{n}/n)$, the optimal utility $O((ne)^{3/5})$ would be no longer achievable.

The above are main steps taken by the user nodes. As for the server node, it simplicity updates the decision variable by adding up partial linear oracles and broadcasts the new decision variable as well as private active features. The algorithm is summarized in Algorithm 2.

### 7.4.2 Analysis

Algorithm 2 comes with guaranteed privacy, utility and uplink communication complexity.

**A. Differential privacy:** The sensitive communicate sourced from user nodes are
the indices and active features, both of which are \((\epsilon/2, \delta/2)\)-differentially private by Lemma D.3.1 and Lemma D.3.2 in the Appendix.

Then, the algorithm is guaranteed to be \((\epsilon, \delta)\)-differential privacy by simple composition property of DP, as summarized by

**Theorem 7.4.1.** Algorithm 2 is \((\epsilon, \delta)\)-differentially private.

**B. Utility:** The utility is based on the convergence result of BCFW-AS developed in the previous section with the sampling being \((K, \tau)\)-distributed sampling. As perturbation and sketching are introduced, the gradient and linear oracle are inexact. The key step is to show Algorithm 2 satisfies Assumption 1 and show that \(\hat{s}^t\) and \(\hat{V}_f(x^t)\) are \(g_l\)-LO and \(g_y\)-PG correspondingly. The following theorem presents the utility guarantee and the parameters appeared are summarized in Table 2, where \(\text{nnz}(a)\) denotes the largest count of nonzero entries of all features, \(p_{\text{min}} = p = \frac{K\tau}{d}\) is the sampling rate, \(\iota\) is the JL-transform parameter and \(G = \frac{1}{n}||A^T(Ax - y)||_\infty = O(1)\).

**Theorem 7.4.2.** Let Assumption 1 and 2 hold. Set \(T = \left(\frac{C^2_{C_B}(K, \tau)^{\nu} n}{C_g + C_l} \right)^{\frac{2}{3}}\). Algorithm 2 ensures the following expected excess empirical risk under \((\epsilon, \delta)\)-differential privacy,

\[
\mathbb{E}[f(x^T)] - \min_{x \in \mathcal{M}} f(x) = O\left(\frac{(C_g + C_l)^{\frac{1}{2}}(C_f^{\frac{2}{3}}(K, \tau))^{\frac{1}{2}} \log(2G\eta K \tau n)}{p^2(ne)^{\frac{2}{3}}}\right). \tag{7.4.16}
\]

**Remark 1.** According to Theorem 7.4.2, the utility is of order \(O\left(\frac{1}{(ne)^{\frac{2}{3}}}\right)\), where \(O(\cdot)\) hides constants and log factors. Compared with the utility result \(O\left(\frac{1}{(m)^{\frac{2}{3}}}\right)\) of the centralized private FW method [129], ours has the same dependence on \(n, \epsilon\) which is nearly optimal for any private algorithm achievable. Our utility is discounted by the sampling rate-related term, which, however, can be regarded as the trade-off between computational scalability and utility.

**C. Uplink communication complexity:** The uplink communication comes from:

1) \(KT\) integers for sending the private index (rather than the entire \(d \times T\) float local decision variables \(x\)); 2) \(m \times KT\) for sending the private active features. Based on \(T\) in Theorem 3 and \(m\) in Table 2, we have
Corollary 7.4.3. Algorithm 2 has uplink communication complexity $O(\frac{1}{e^2} \log(n/e)^{1/3} K (n/e)^{2/3})$.

Remark 2. Compared to the one-shot communication at the preprocessing proposed by method $[53]$, our uplink communication cost has better dependence on the sample size with $O(n^{2/3} \log(n)^{1/3})$ than theirs with $O(nr \log r)$, which shows our “share-at-need” feature sharing is more efficient than random sketching at preprocessing.

7.5 Summary

We have considered private training of ERM model with the features distributed among user nodes and developed an uplink communication efficient, utility nearly optimal algorithm based on a new general analysis of BCFW algorithm under arbitrary sampling. Under the same privacy budget, our distributed variant has: 1) the same order of nearly optimal utility guarantee for the LASSO task as centralized counterpart, and 2) improved overall uplink communication complexity than existing method for the same feature-distributed setting. To derive the convergence analysis, we have introduced a universal step size as a new expected curvature notion, which comes with the detailed comparison with existing quantities. We have demonstrated the flexibility of the convergence analysis by recovering exact, refined or matchable result of existing BCFW method under specific samplings.
Algorithm 19 Differentially Private Frank-Wolfe Algorithm with Distributed Features for LASSO

Server Node

Input: Initial feasible variable $x_0$, step sequence $\gamma^t$, maximum iteration $T$, $\ell_1$-norm ball size $\eta$;
1: for $t = 0, 1, \ldots, T - 1$ do
2: Receive $\hat{idx}^t_k$ and $\hat{a}^t_k$ from $K$ workers;
3: $\hat{s}^t = \eta \sum_{k=1}^{K} -\text{sign}(\hat{idx}^t_k)e_{[\hat{idx}^t_k]}$;
4: $x^{t+1} = (1 - \gamma^t)x^t + \gamma^t \hat{s}^t$;
5: $\hat{q}^t = (1 - \gamma^{t-1})q^{t-1} + \gamma^{t-1} \sum_{k=1}^{K} -\text{sign}(\hat{idx}^t_k)\frac{1}{\eta}\hat{a}^{t-1}_{[\hat{idx}^t_k]}$;
6: Broadcast $x^{t+1}$ and $\hat{q}^t$ to all $K$ workers;
7: end for

Output: $x^T$;

User Node

Input: sampling parameter $\tau$, JL-transform matrix $J$
7: for $t = 0, 1, \ldots, T - 1$ do
8: Receive $x^t$ and $\hat{q}^t$ from server;
9: Sample $\tau$ coordinates $T_k^t \subseteq P_k$ randomly;
10: For each $i \in T_k^t$, $\hat{\nabla}_i f(x^t) = a_i^T \hat{q}^t - a_i^T y$, $v_i = \hat{\nabla}_i f(x^t) + \text{pert}$;
11: $\hat{idx}^t_k = \text{sign}(v_i) \arg \max_{i \in T_k^t} v_i$;
12: Send $\hat{idx}^t_k$ and $\hat{a}^t_k = J a_{[\hat{idx}^t_k]} + \xi$ to server;
13: end for
Chapter 8

Differentially Private Streaming Convex Optimization with Decayed Privacy

8.1 Introduction

In many practical applications, data are collected and fed to analyzing algorithms in a streaming fashion. Prominent examples can vary from online user behavior analysis with ever increasing social media data, website cookies and user click history, to monitoring systems that continuously collect sensory data for applications like personal trajectory monitoring and location-based services, to many other real-time systems used for industrial manufacture or financial transaction data analysis. Constraint online convex optimization (COCO) is a popular approach to dealing with various streaming machine learning and data mining applications that raise in such real-time learning and monitoring systems environment. Within COCO context, with data instances arriving sequentially over time, the COCO algorithm responses continuously by releasing new decision variable from a bounded convex set at each timestamps. Each time the COCO algorithm plays a variable, it observes a convex loss and the aim is to minimize the regret, a utility measure for COCO.
algorithms.

The outputs of the learning algorithm can put the privacy of sensitive data at risk, which is especially severe for online systems as an adversary can potentially observe the entire response sequence that contains significantly more information than only one final output produced by an offline system. To preserve privacy of individual data instances, the rigorous statistical notion of differential privacy (DP) \cite{32, 33} has become nowadays the standard technique for privacy guarantee. DP restricts that any single change of the data instances will make little difference to the output of a random algorithm. The utility of the algorithm often decreases due to the additional randomness introduced for privacy protection. As such, algorithm designers need to strike the trade-off between privacy and utility, which becomes even more delicate for online algorithm with the continuing release of output sequence. With streaming data accumulating, it becomes difficult to protect each instance’s privacy without much compromise on utility. Some works propose to maintain satisfactory utility by relaxing the DP restriction on some instances or events. For example, \cite{69} considers \(w\)-event privacy to only protect the privacy of events of every \(w\) continuous timestamps in a sliding window, while \cite{18} considers \(l\)-trajectory privacy that only ensures differential privacy of every length \(l\) trajectory. In particular, \cite{9} observes that recent data are more important than distant inputs and therefore proposes the privacy expiration assumption by protecting only the privacy of recent data instances inside a sliding window. However, these are results within database community that publishes either private data vectors or simple statistics like sum or histogram. It has yet to be known whether similar relaxation of privacy on some instances can improve the performance of private COCO methods.

Within convex learning, despite the various offline DP algorithms \cite{21} that adapt to different problem structures, e.g. private gradient descent for unconstrained function \cite{21}, private projected gradient (PG) descent for projection-friendly constraint sets \cite{3}, or private conditional gradient (CG) for linear oracle-friendly constraint sets \cite{129}, private COCO algorithms \cite{61, 130, 63} are relatively less studied. With
increasing number of instances, the existing private COCOs accumulate regret faster than nonprivate optimal counterparts, which inspires us the growing price of utility to pay for privacy protection for every streaming-in individual instance over time. Also, they rely exclusively on the projected gradient (PG) step for updating the variable provided that the constraints are projection-friendly, regardless of the existence of many widely applied models yet with linear oracle-friendly constraint sets, i.e. suiting a CG step better than a PG step, see [60]; [42].

In this chapter, we therefore aim at improving the performance of the private COCO algorithm by trading the privacy of distant data under the privacy expiration assumption [9]. To take the advantage of the relaxed privacy restriction on distant data timely and safely, we adapt the window tree mechanism with either Gamma or Gaussian perturbation to maintain a private gradient summation approximation satisfying the window differential privacy. We then utilize the popular Follow The Approximate Leader (FTAL) scheme to update the response variable by optimizing an approximate function constructed with the private gradient summation. By virtue of the flexibility of the window tree mechanism for the private gradient summation estimation, we are able to show two variants: the first solves the approximate function exactly by projection onto the constraint set, while the second variant solves it approximately by a variant of the conditional gradient step. With either update under corresponding assumptions, we are able to achieve improved regret growth of order $O(\ln T)$ with respect to the sequence length $T$, which matches the nonprivate optimal regret bound. The regret bound also captures the effect of the window length and provides explicit trade-off between privacy and utility. In particular, our second variant, to the best of our knowledge, is the first private COCO algorithm to consider a CG update in the private COCO, providing an alternative choice for private online learning tasks on linear oracle-friendly constraint sets.

In brief, we have the following contributions: 1) Reduce regret to $O(T)$ for private COCO by trading privacy of remote inputs, so that the utility matches the same order of the nonprivate counterparts; 2) For privacy-protection window size $W$, the
regret has a factor \( \ln(W) \), which captures the trade-off between privacy coverage and utility; 3) Provide a projection-free COCO variant, which is the first private conditional gradient method under online setting.

8.2 Preliminary

8.2.1 Problem Setup

We formally describe our problem setting. Recall we are given a streaming sequence of loss functions \( \mathcal{I} = [f_1, f_2, ..., f_t, ... f_T] \) arriving one at a time, the COCO algorithm is required to response \( x_t \) from the constraint set \( \mathcal{C} \), which is a bounded convex set. After each response, it will suffer a convex loss \( f_t(x_t) \). This chapter assumes that \( f_t(x) \) is \( L \)-Lipschitz continuous and \( \mu \)-strongly convex w.r.t. \( x \). We also assume that the bounded convex constraint set has the diameter \( D \), i.e. \( D = \max_{x, y \in \mathcal{C}} ||x - y||_2 \).

To enable a deterministic algorithm to provide DP protection, it is common to randomize it by adding designed random noise values sampled from proper random distributions, including Laplacian, Gamma, Gaussian and more recently the Wishart distribution \([65]\). The sensitivity of the algorithm is an important concept in deciding the parameter of the random distribution, which measures the maximum change of the algorithm when a particular individual instance changes.

**Definition 8.2.1.** (\( \ell_2 \) Sensitivity) Let \( \mathcal{A} \) be an algorithm mapping stream \( \mathcal{I} \) to \( \mathcal{O} \). The \( \ell_2 \) sensitivity of \( \mathcal{A} \) is \( \Delta_2 = \max_{\mathcal{I}, \mathcal{I}'} ||\mathcal{A}(\mathcal{I}) - \mathcal{A}(\mathcal{I}')||_2 \), where \( \mathcal{I} \) and \( \mathcal{I}' \) differ only in one instance.

8.2.2 Differentially Private Follow The Approximate Leader

The private COCO algorithms \([61]\) and \([130]\) are based on the popular nonprivate online learning scheme Follow The Approximate Leader (FTAL) \([119, 49]\), which only requires the first-order information and is known to be regret optimal. For \( \mu \)-strongly convex loss function \( f_\tau(x) \), it updates \( x_{t+1} \) by minimizing an approximation function.
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Privacy guarantee</th>
<th>per-iteration OP</th>
<th>Regret Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>POCG [61]</td>
<td>$(\epsilon, \delta)$-DP</td>
<td>two projections</td>
<td>$O\left(\frac{\sqrt{d} \ln^2 T}{\epsilon}\right)$</td>
</tr>
<tr>
<td>PQFTL [61]</td>
<td>$(\epsilon, \delta)$-DP</td>
<td>linear equation</td>
<td>$O\left(\frac{\sqrt{d} \ln \frac{1}{\epsilon} \ln^{1.5} T}{\epsilon}\right)$</td>
</tr>
<tr>
<td>PFTAL [130]</td>
<td>$\epsilon$-DP</td>
<td>one projection</td>
<td>$O\left(\frac{d \ln^{2.5} T}{\epsilon}\right)$</td>
</tr>
<tr>
<td>Ours with projection window</td>
<td>$\epsilon$-DP</td>
<td>one projection</td>
<td>$O\left(\frac{d \log^{1.5} W \ln^{1.5} T}{\epsilon}\right)$</td>
</tr>
<tr>
<td>Ours with projection window $(\epsilon, \delta)$-DP</td>
<td>one projection</td>
<td>$O\left(\frac{d \log^{1.5} W \ln^{1.5} T}{\epsilon}\right)$</td>
<td></td>
</tr>
<tr>
<td>Ours with linear window</td>
<td>$\epsilon$-DP</td>
<td>one linear oracle</td>
<td>$O\left(\frac{d \log^{1.5} W \ln^{1.5} T}{\epsilon}\right)$</td>
</tr>
<tr>
<td>Ours with linear window $(\epsilon, \delta)$-DP</td>
<td>one linear oracle</td>
<td>$O\left(\frac{d \log^{1.5} W \ln^{1.5} T}{\epsilon}\right)$</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.1: Comparison of private COCO algorithms

of the streaming empirical risk minimization function $\sum_{\tau=1}^{t} f_\tau(x)$ (a.k.a. Follow The Leader (FTL) updating) as

$$F_t(x) = \sum_{\tau=1}^{t} \tilde{f}_\tau(x), \text{ where } \tilde{f}_\tau(x) = f_\tau(x_\tau) + \langle \nabla f_\tau(x_\tau), x - x_\tau \rangle + \frac{\mu}{2} \|x - x_\tau\|^2, (8.2.1)$$

which is always a lower approximation by the strong convexity assumption. To minimize $F_t(x)$, it is equivalent to minimize

$$J_t(x) = \langle \sum_{\tau=1}^{t} \nabla f_\tau(x_\tau), x \rangle + \frac{\mu}{2} \sum_{\tau=1}^{t} \|x - x_\tau\|^2. \quad (8.2.2)$$

By observing that the continuous gradient summation $g_t = \sum_{\tau=1}^{t} \nabla f_\tau(x_\tau)$ is the only ingredient having access to sensitive private information in the above procedure, it suffices to use a differentially private surrogate of $g_t = \sum_{\tau=1}^{t} \nabla f_\tau(x_\tau)$ in the computation to ensure the entire COCO algorithm is DP as well. [61, 130] abstract out the private gradient summation maintenance sub-step out as a continuous running sum task and resort to the tree mechanism. The tree mechanism [34, 20] has been proven to add minimum noise to ensure DP and can be implemented in a space and computational efficient manner, is therefore the ideal candidate for maintaining the private gradient sum. Denote the private surrogate of gradient summation computed by the tree mechanism by $\hat{g}_t$, and re-denote the private release history of the private COCO up to timestamp $t$ by $\hat{x}_1, \hat{x}_2, ..., \hat{x}_t$, the private updating replaces $g_t$ with $\hat{g}_t$.
to minimize

\[
\hat{J}_t = \langle \hat{g}_t, x \rangle + \frac{\mu}{2} \sum_{\tau=1}^{t} \|x - x_\tau\|_2^2.
\]  

(8.2.3)

[61] and [130] propose to exactly solve it by projection \( \mathcal{PO} \): \( \hat{x}_{t+1} = \mathcal{PO}(\sum_{\tau=1}^{t} \hat{x}_\tau - \hat{g}_t/\mu, \mathcal{C}) \), which potentially limits the their applicability to only tasks with projection-friendly constrains, i.e. constraint sets where projection can be efficiently evaluated, like Euclidean ball.

\[8.3\] Proposed Method

This section presents our private COCO algorithm with improved regret bound by considering the privacy expiration assumption. Following the FTAL framework, we start with a window tree mechanism that maintains two binary subtrees with depth at most \( \lceil \log_2 W \rceil + 1 \) to keep the window private gradient summation for constructing \( \hat{J}_t(x) \) in eq. (8.2.3). In subsection 8.3.2, we solve \( \hat{J}_t(x) \) exactly by projection for projection-friendly constraints, which gives us a direct comparison with previous [61] [130] methods without privacy expiration assumption (please see Table 8.1 for comparison). The regret analysis shows the benefit of privacy expiration in improving regret bound to \( O(\ln T) \) given the fixed window size, matching the nonprivate optimal w.r.t. sequence length \( T \). Also, it captures the trade-off between the privacy coverage and the regret bound. The window tree mechanism based privacy with expiration gradient sum approximation technique is flexible enough that, for the first time in the context of private COCO, in subsection 8.3.3 we further develop a private version of online CG based method for linear oracle-friendly constraints, which still maintains the optimal \( O(\ln T) \) regret. We relegate additional contents such as the window tree mechanism description and proofs of theorems in this section to the Appendix.
8.3.1 Window Tree Mechanism for Private Gradient Summation with Decayed Privacy

The window tree mechanism \([9]\) is an adaptation of the tree mechanism with consideration on the privacy expiration assumption, which has been extensively used for private running sum estimation. We use it as a building block to keep the sum of gradients. For the sake of space, we only describe the privacy and utility results of the window tree mechanism in Subsection 3.1.1 and 3.1.2, while relegating the detailed introduction of the window tree mechanism (denoted by \(WTM\)) to the Appendix and also referring readers to [20] and [9] for rigorous algorithm description.

Window Tree Mechanism with Gamma Perturbation

We consider the window tree mechanism for general streaming data \(v_1, v_2, \ldots, v_t, \ldots\) to preserve \(\epsilon\) differential privacy with the window size \(W\) for releasing summation sequence \((s_1, s_2, \ldots, s_t, \ldots)\). We denote the noise vectors added to the nodes in the two rightmost subtrees (explained in the Appendix) by \(n_i\). The next lemma describes the distribution parameter of \(n_i\) and shows the window \(\epsilon\)-DP guarantee with Gamma perturbation denoted by \(\Gamma\).

**Lemma 8.3.1.** With \(n_i \in \mathbb{R}^d\) sampled to satisfy \(\|n_i\|_2 \sim \Gamma(d, \frac{\Delta_2([\log_2 W]+1)}{\epsilon})\) (i.e. \(\|n_i\|_2\) proportional to \(e^{-\frac{\|n_i\|_2^2}{\Delta_2([\log_2 W]+1)}}\) and added to each relevant node, the window tree mechanism achieves window \(\epsilon\)-differential privacy with window size \(W\).

The next lemma describes the utility of the window tree mechanism in terms of the \(\ell_2\)-norm distance between the private and clean running sums. Note that in general we are interested in the case \(t > W\) and leave along the \(t \leq W\), which is identical to the original tree mechanism.

**Lemma 8.3.2.** For any \(\beta > 0\) and \(t\), with probability at least \(1 - \beta\), \(s_t\) computed by window tree mechanism with Gamma noise at each node of \(\|n_i\|_2 \sim \Gamma(d, \frac{\Delta_2([\log_2 W]+1)}{\epsilon})\) satisfies:

\[
\|s_t - \sum_{i=1}^{t} v_i\|_2 \leq \left( \frac{d \Delta_2 \log_2 1.5^\beta W \ln \frac{d}{\beta}}{\epsilon} \right),
\]
where $\Delta_2$ is the $\ell_2$ sensitivity of the sum function.

**Window Tree Mechanism with Gaussian Perturbation**

Similarly, to ensure window $(\epsilon, \delta)$-DP ($\delta > 0$), we add Gaussian noise to the nodes of the subtrees involved in the privacy window.

**Lemma 8.3.3.** With noise vector $n_i \sim \mathcal{N}(0, \sigma^2 I_{d \times d})$, where $\sigma^2 = \frac{8\Delta^2_2([\log_2 W]+1)^2 \ln(2/\delta)}{\epsilon^2}$, and added to each relevant node, the window tree mechanism achieves window $(\epsilon, \delta)$-differential privacy with window size $W$.

Similarly, we provide the utility bound.

**Lemma 8.3.4.** For any $\beta > 0$ and $t$, with the probability at least $1 - \beta$, $\bar{s}_t$ computed by window tree mechanism satisfies:

$$||s_t - \sum_{i=1}^{t} v_i||_2 = O(\frac{\sqrt{d\log_2^1 5 W \ln \frac{1}{\beta} \sqrt{\ln \frac{1}{\beta}}}}{\epsilon}),$$

(8.3.5)

where $\Delta_2$ is the $\ell_2$ sensitivity of the sum function.

Later, we will show that, with Gaussian perturbation, the compromise on privacy with a small probability $\delta$ can improve the regret bound dependence of the dimension $d$ in private COCO from $O(d)$ to $O(\sqrt{d})$, when compared with Gamma perturbation based methods like PFTAL in [130]. That is, the private COCO can inherit the improved utility of Lemma 8.3.4.

### 8.3.2 Window Differentially Private COCO with Projection

We follow the FTAL framework and provide window privacy protection by utilizing the private approximation of the gradient summation maintained by window tree mechanism presented in the previous subsection. That is, at the $t$-th timestamp, we update the response $\hat{x}_{t+1}$ by exactly solving $\hat{J}_t(x)$ with $\hat{g}_t$ computed by window tree mechanism. We call the proposed algorithm **Window Differentially Private COCO with Projection (WDP-COCOP)** and summarize it in Algorithm 20. Step 2 has two
options to add either Gamma or Gaussian perturbation to the relevant nodes in the window tree, depending on the privacy type (i.e. $\epsilon$ or $(\epsilon, \delta)$). The regret bound lends us direct comparison with previous algorithms based on similar building blocks but without privacy expiration assumption, clearly revealing the effect of the window differential privacy in COCO as a result.

**Algorithm 20** Window Differentially Private COCO with Projection: WDP-COCOP

**Input:** $x_1 = \hat{x}_1, T, W, \mu, L, \epsilon$ or $(\epsilon, \delta)$, loss function sequence $f_1, f_2, ..., f_T$.

1: for $t = 1, 2, ..., T$ do
2: Construct $\hat{g}_t$ by window tree mechanism $WTM$:
   - **OPTION I:** $WTM_{\text{Gamma}}(\nabla f_t(\hat{x}_t), W, \epsilon)$ for window $\epsilon$-DP;
   - **OPTION II:** $WTM_{\text{Gaussian}}(\nabla f_t(\hat{x}_t), W, \epsilon, \delta)$ for window $(\epsilon, \delta)$-DP;
3: $\hat{x}_{t+1} = PO((\sum_{\tau=1}^{t} \hat{x}_\tau - \hat{g}_t/\mu), C)$;
4: **Output:** $\hat{x}_{t+1}$;
5: end for

**Privacy Guarantee:**

The WDP-COCOP algorithm preserves window differential privacy with proper noise parameters as described by the following two theorems. In brief, they are by Lemma 8.3.1 & 8.3.3 with $\ell_2$ sensitivity being $(L + \mu D)$, plus the post-processing property of differential privacy (Proposition 2.1 in [35]).

**Theorem 8.3.5.** *(window $\epsilon$-DP of Algorithm 20)* Let $f_1, ..., f_T$ be streaming $L$-Lipschitz continuous and $\mu$-strongly convex functions. The diameter of the bounded convex set is $D$. Algorithm WDP-COCOP with OPTION I is window $\epsilon$-differential privacy with window size $W$ if window tree mechanism samples Gamma noise from distribution $||n_t||_2 \sim \Gamma((\frac{L+\mu D}{\epsilon}(\log_2 W + 1))$.

**Theorem 8.3.6.** *(window $(\epsilon, \delta)$-DP of Algorithm 20)* Let $f_1, ..., f_T$ be streaming $L$-Lipschitz continuous and $\mu$-strongly convex functions. The diameter of the bounded
convex set is $D$. Algorithm WDP-COCOP with OPTION II is window $(\epsilon, \delta)$-differential privacy with window size $W$ if window tree mechanism samples Gaussian noise from distribution $n_i \sim \mathcal{N}(0, \frac{8(L+\mu D)^2(\log_2 W)^2 + 1}{\epsilon^2} \ln^2(2/\delta) \mathbb{I}_{d \times d})$.

**Regret Analysis:**

The regret analysis for Algorithm 1 WDP-COCOP with both privacy options are provided.

**Theorem 8.3.7.** *(regret guarantee with window $\epsilon$-DP)* Under the same condition as in Theorem 1, for any $\beta > 0$ and window size $W$, with probability at least $1 - \beta$, the following regret bound holds,

$$
\text{Regret}(T) = O\left(\frac{d(L + \mu D)^2 \log^{1.5} W \ln \frac{d}{\beta} \ln T}{\mu\epsilon}\right).
$$

**Theorem 8.3.8.** *(regret guarantee with window $(\epsilon, \delta)$-DP)* Under the same condition as in Theorem 2, for any $\beta > 0$ and $W$, with probability at least $1 - \beta$, the following regret bound holds,

$$
\text{Regret}(T) = O\left(\frac{\sqrt{d}(L + \mu D)^2 \log^{1.5} W \ln \frac{1}{\beta} \sqrt{\ln \frac{1}{\beta} \ln T}}{\mu\epsilon}\right).
$$

**Discussion:**

Our method shares similar building blocks as PQFTL [61] and PFTAL [130], including tree mechanism, FTAL and projection. However, they have not explored the privacy expiration assumption and it is also unclear how much privacy budget one has to assign for protecting certain amount of instances there. In fact, with fixed window size $W$, our regret bound is of order $O(\ln T)$, which is better than theirs and is also known to match the nonlinear optimal. Also, PQFTL [61] is limited to quadratic loss functions. Taking window size $W$ related term into comparison, our regret bounds provide explicit trade-off between privacy coverage and regret bound, i.e. the regret scales with the sliding window size. Compared to the more generally applicable PFTAL [130], even with full sequence privacy protection (i.e. $W = T$), our method with Gaussian perturbation (option II) has better dependence on $d$ (i.e. $O(\sqrt{d})$ to
Algorithm 20 computes the exact minimum of \( \hat{J}_i(x) \) by projection, which is adopted by almost all existing private COCO methods provided that the constraints are projection-friendly. This, however, limits the applicability of delivering privacy protection of COCO algorithms for other widely applied applications with linear oracle-friendly constraints, which will be further addressed in the following.

### 8.3.3 Window Differentially Private COCO with Linear Oracle

We present a private COCO called **Window Differentially Private COCO with Linear oracle** (WDP-COCOL) in Algorithm 21 targeting at tasks with linear oracle (LO)-friendly constraint sets, for example matroid polytope, flow polytope, spectrahedron and many others [60]. The conditional gradient method, also known as the Frank-Wolfe method, has become increasingly popular for convex optimization over LO-friendly constraint sets, where either the improved computational efficiency or the better ability to preserve the sparse structure of the desired variable when computed with the CG methods have been observed [60]. In particular, most recently, [129] provides a CG-based private algorithm in the offline empirical risk minimization setting and shows that it achieves nearly optimal utility for Lasso regression under DP restriction. [51, 179, 42] consider online CG methods, however in the nonprivate setting, it is still unclear whether CG can be carried out in a private manner within the streaming context.

We study private COCO with the CG step based on the nonprivate FTAL type conditional gradient variant [42] and consider constraint set to be general bounded polytope \( \mathcal{P} \). It uses a slightly different approximation function \( F_t(x) \) with eq. (8.2.1):

\[
F_t(x) = \sum_{r=1}^t \tilde{f}_r(x) + \frac{C_m}{2} \|x - x_1\|_2^2.
\]

Thus our modified private \( \hat{J}_t(x) \) is \( \hat{J}_t = \langle \hat{g}_t, x \rangle + \frac{\mu}{2} \sum_{r=1}^t \|x - \hat{x}_r\|_2^2 + \frac{C_m}{2} \|x - x_1\|_2^2 \), where \( \hat{g}_t \) can be maintained again by window tree mechanism. The key ingredient to the CG based method is a linear
oracle evaluation step. Recently, various modifications have been proposed based on the original LO. This chapter, in specific, follows [42] to use the local linear oracle, denoted by $\mathcal{LLOO}$, which returns a linear oracle $p$ given variable $x$ and gradient $g$. Roughly, $\mathcal{LLOO}$ chooses a linear oracle from a smaller range that $p$ is in the $\rho \cdot r$ ball centered around $x$ (denoted by $\mathbb{B}(x, r)$), rather than from the entire set $\mathcal{P}$. [42] shows that on general polytopes, $\mathcal{LLOO}$ still computes one local LO evaluation per-timestamp. The functioning of $\mathcal{LLOO}$ is summarized in the following assumption.

**Assumption 8.3.1.** Denote the local linear oracle by $\mathcal{LLOO}$ that for variable $x$, radius $r$ and gradient estimation $g$. It evaluates the linear oracle $p \in \mathcal{P}$, i.e. $p = \mathcal{LO}(x, r, g)$. We assume the linear oracle $p$ satisfies the following two properties.

1. $\forall y \in \mathbb{B}(x, r) \cap \mathcal{P}$, $\langle p, g \rangle \leq \langle y, g \rangle$; 2. $||x - p||_2 \leq \rho r$.

In Algorithm 21, Lines 6 sets the radius $r_t$; Step 7 computes the private local linear oracle $\hat{p}_t$; Line 8 computes the new private decision variable $\hat{x}_{t+1}$ by linear combination, which is common to CG based method. Although our method is based on nonprivate [42], the design of parameters and analysis are different. For example, the settings of $C_1, C_W$ take the window size and DP parameters $\epsilon, \delta$ into consideration, which are exclusive to window differential privacy. Also, the regret analysis needs to handle the approximate linear oracle output of $\mathcal{LLOO}$ because of the randomness inherited in $\hat{g}_t$ for privacy, rather than the deterministic clean $g_t$ as considered in [42].

**Privacy Guarantee:**

The privacy guarantee is similar to the previous subsection, just to realize that the $\mathcal{LLOO}$ and linear combination steps are all post-processing of the private gradient summation, which does not affect the differential privacy guarantee.

**Theorem 8.3.9.** (*window differential privacy of Algorithm 21*) Under the same condition as in Theorem 8.3.5 and 8.3.6 (with $\mathcal{C}$ replaced by $\mathcal{P}$), Algorithm 21 with
Algorithm 21 Window Differentially Private COCO with Linear Minimization: WDP-COCOL

**Input:** \( x_1 = \hat{x}_1, T, W, \mu, \beta, L, c, d, \epsilon \) or \((\epsilon, \delta)\), loss function sequence \( f_1, f_2, …, f_T \).

1: initialize: \( \rho = c\sqrt{d} \) (c is a geometry constant of \( P \)), \( \alpha = \frac{1}{\rho^2}, C_0 = (25\rho^2)^2 \);
2: initialize (cont.): \( C_W = \frac{2\alpha d \log_2^2 W \ln \frac{d}{\epsilon}}{(1-\alpha)\epsilon} \) (for \( W\)-DP); or \( C_W = \frac{2\alpha d \log_2^2 W \ln \frac{1}{\delta} \ln \frac{d}{\epsilon}}{(1-\alpha)\epsilon} \) (for \( W\)-(\( \epsilon, \delta \))-DP);
3: initialize (cont.): \( C_1 = 26\rho^2(L + \mu D)(1 + C_W) \);
4: for \( t = 1, 2, …, T \) do
5: Construct \( \hat{g}_t \) by window tree mechanism \( \text{WTM} \):
   
   **OPTION I:** \( \text{WTM}_{\text{Gamma}}(\nabla f_t(\hat{x}_t), W, \epsilon) \) for window \( \epsilon \)-DP;
   
   **OPTION II:** \( \text{WTM}_{\text{Gaussian}}(\nabla f_t(\hat{x}_t), W, \epsilon, \delta) \) for window \((\epsilon, \delta)\)-DP;
6: \( \eta_t = \frac{C_t^2}{\mu(t+C_0)}; r_t = \sqrt{\frac{4\eta_t}{\mu(t+C_0)}} + \frac{2(L+\mu D)}{\mu(t+C_0)} \);
7: \( \hat{p}_t = L\text{LDO}(\hat{x}_t, r_t, (\sum_{t=1}^{t} \hat{x}_t + C_0 x_1 - \hat{g}_t)\mu) \);
8: \( \hat{x}_{t+1} = (1-\alpha)\hat{x}_t + \alpha\hat{p}_t \);
9: **Output:** \( \hat{x}_{t+1} \);
10: **end for**
Option I and Option II is window $\epsilon$-differential privacy and window $(\epsilon, \delta)$-differential privacy with window size $W$ correspondingly.

Regret Analysis:

**Theorem 8.3.10.** (regret guarantee with window $\epsilon$-DP) Under the same condition as in Theorem 8.3.9, for any $\beta > 0$ and window size $W$, with probability at least $1 - \beta$, Algorithm 21 with OPTION I has the following regret bound at a particular iteration $T \geq W$, 

\[
\text{Regret}(T) = O\left(\frac{d(L + \mu D)^2 \log^{1.5} W \ln \frac{d}{\beta} \ln T}{\mu \epsilon}\right).
\]  

(8.3.8)

**Theorem 8.3.11.** (regret guarantee with window $(\epsilon, \delta)$-DP) Under the same condition as in Theorem 8.3.11, for any $\beta > 0$ and window size $W$, with probability at least $1 - \beta$, Algorithm 21 with OPTION II has the following regret bound with sequence length $T \geq W$, 

\[
\text{Regret}(T) = O\left(\max\left(\sqrt{d\epsilon}, \log_2^{1.5} W \ln \frac{1}{\delta} \sqrt{\ln \frac{1}{\beta}} \right) \cdot \frac{\sqrt{d(L + \mu D)^2 \ln T}}{\mu \epsilon}\right).
\]  

(8.3.9)

Discussion:

The regret in this part for both privacy types matches nonprivate optimal $O(\ln T)$ with the fixed window size, confirming that online CG can indeed be privatized and broaden the applicability of private COCO to many applications with LO-friendly constrains. Again, the term $\log_2^{1.5} W$ captures the trade-off between privacy coverage and utility with the regret bound scaling with a polylog term of window size $W$. Considering $\epsilon$-DP, our regret bound is as good as $\text{[130]}$ even when privacy window extends to the full sequence. When relaxing to $(\epsilon, \delta)$-DP, our method can have better dependence on the dimension $d$ by Theorem 8.3.11.

8.4 Summary

In this chapter, we have investigated the potential to trade privacy of remote input instances for better utility in the context of private COCO. We have adapted a window
tree mechanism with Gamma and Gaussian perturbation to efficiently maintain the window private gradient summation. We have provided two variants WDP-COCOP and WDP-COCOL, to preserve window DP with provable privacy guarantee and regret bound. The regret bounds of both have shown that the proposed algorithms can adaptively adjust the trade-off between privacy coverage and utility. Meanwhile, with the fixed window size, the regret bound growth matches nonprivate optimal $O(\ln T)$. In particular, WDP-COCOL can be of independent interest as the first private COCO method designed for problems with linear oracle-friendly constraints, effectively broadening the applicability of private COCO methods.
Chapter 9

Conclusions

9.1 Conclusions

Sparse and low rank modelings are a fundamental tools in machine learning and find wide applications in computer vision, data mining, bioinformatics and so on. Their associated sparse/low rank optimization problems become important for efficiently making use of these modeling methods. To develop more optimal algorithm, especially to take into consideration of computational, communication and privacy restrictions, become the key to algorithm design.

In this thesis, I made the following contributions: Firstly, I presented two novel algorithms to improve the gradient computational efficiency upon the corresponding current methods.

- For generalized conditional gradient methods, the inferior iteration complexity incurs excess number of gradient evaluation, which can counteract the efficiency gained by low cost linear subproblem. I therefore propose a novel algorithm that requires optimal count of graduate evaluations as proximal gradient. I also propose a refined variant for a type of gauge regularized problem where approximation techniques are allow to further accelerate linear subproblem computation. Experiments of an group sparse regularized CUR-like matrix factorization problem on four real-world datasets demonstrate the efficiency of
the proposed method.

- For proximal gradient methods, based on the fact that incremental gradient methods, which exploits the finite sum structure of the loss function, have been shown that converge as fast as full gradient methods while have loss per-iteration cost as stochastic gradient method. Moreover, composite penalties such as group lasso and graph lasso are applied to regularize the ERM model to induce structured sparsity. Such complex nonsmooth regularizer can make the problem hard to optimize. Proximal average is proposed as a better approximation than smoothing method and a more compact alternative approach than ADMM to handle such regularizer. Inspired by both of the recent advances, I then propose a new algorithm to efficiently solve ERM problem with composite regularizer. This method is scalable to large scale problems.

Then, I provided two novel methods to improve the per-iteration computational complexity in robust low rank matrix and tensor optimization.

- For the matrix case, I studied the robust low rank matrix learning task with spectral k-support norm being the low rankness regularizer. My method can utilize both the squared spectral k-support norm and itself. For both formulations, I converted the objective to the dual equivalence, which required to computer only a linear subproblem per-iteration. Therefore, it avoided expensive proximal mapping of the spectral k-support norm. Furthermore, by studying the (sub)gradient of the dual norm of the more generalized spectral k-support norm, I have incorporated a line search strategy that is able to adapt to smoothness change. Experiment result on both synthetic and real computer vision applications of background modeling and face reconstruction have successfully demonstrated the superiority of the proposed method in comparison with the existing counterparts.

- For the tensor case, I have studied the robust low-rank tensor minimization problem under the t-SVD framework. After deriving an alternative relaxation
to the sum of tubal multi-rank by providing a novel tensor spectral $k$-support norm, I have shown that TSP-$k$ norm interpolates between TNN and tensor Frobenius norm, which is helpful for preserving more global information of the intrinsic low-rank tensor. Then, I provided two optimization methods for dealing with both medium and larger scale data, based on the primal and dual form of the objective function, correspondingly. Experiments on synthetic, as well as image and video applications on both medium and larger-scale dimension datasets verified the superiority of TSP-$k$ over TNN for low rank tensor modeling as well as the effectiveness of the two proposed optimization methods for their targeted data scales.

Finally, under differential privacy restriction, I proposed two new ERM sparse optimization methods with two common scalable settings of distributed and streaming, correspondingly.

- For the distributed setting, I focused on a rarely studied differentially private features-wise distributed dataset context and developed an uplink communication efficient, utility nearly optimal algorithm based on a new general analysis of BCFW algorithm under arbitrary sampling. Given the same privacy budget, my distributed variant featured two strengths over known method: 1) ensuring a nearly optimal utility guarantee for the private LASSO task as its centralized counterpart, which means the scalability improvement did not compromise the utility of the learned model; 2) improving overall uplink communication complexity. New theoretical techniques has been developed to obtain the much general convergence result that I have introduced a universal step size as a new expected curvature notion, which come with the detailed comparison with existing quantities. I have demonstrated the flexibility of the convergence analysis by recovering exact, refined or matchable result of existing BCFW method under specific samplings.

- For the streaming setting, I proposed to trade privacy of outdated input samples for better utility for private streaming convex optimization. I adapted a window
tree mechanism with Gamma and Gaussian perturbation to efficiently maintain the window private approximation of the gradient summation. I have considered two variants of WDP-COCOP and WDP-COCOL, to preserve window DP with provable privacy guarantee and regret bound. I have shown that The regret bounds can adaptively adjust the trade-off between privacy coverage and utility. At the same time, the regret bound growth matched the nonprivate counterpart of the optimal $O(\ln T)$ rate, provided the window size is fixed. In particular, WDP-COCOL can be of independent interest because it is the first private COCO method suited to linear oracle-friendly constraints, effectively broadening the applicability of private COCO methods.

9.2 Future Work

In my future work, some issues may be further explored along three-fold directions:

1. Distributed adaptation of the proposed gradient-efficient algorithms: In Chapter 3 and 4, I have proposed two gradient-efficient optimization methods under centralized setting. One area of further research is to consider the decentralized variants, which will be useful for larger datasets of which the computation and storage cannot be accommodated on a single computing node.

2. More applications for robust low rank matrix and tensor with spectral $k$-support norm regularization: In Chapter 5 and 6, we applied the robust low rank matrix/tensor algorithms to computer vision and image/video processing applications like background modeling, face reconstruction, image/video denoising. It is interesting to consider a wider range of applications in areas like data mining, recommending system, bioinformatics and so on.

3. Differentially private algorithms for low rank constrained model optimization: In Chapter 7 and 8, I studied communication efficiency and sample efficiency for distributed and streaming private learning under sparse constraints, correspondingly. It is worth considering how to design privacy protection mechanisms for
more general regularization types like low rank constraints, especially when practical design issues like communication cost, utility performance are taken into consideration.
Bibliography


171


Appendix A

Proofs of Materials in Chapter 3

A.1 Proof of Theorem 3.3.1 and Corollary 3.3.2

A.1.1 Optimal FO Evaluation

This proof borrows techniques from [81] and [144].

To begin with, recall the following equation,

\[
\langle x - y, x - z \rangle = \frac{1}{2}||x - y||^2 + \frac{1}{2}||x - z||^2 - \frac{1}{2}||y - z||^2, \forall x, y, z \in \mathbb{X}.
\]  \hspace{1cm} (A.1.1)

By steps 2 and 5 of the algorithm GCG-GS,

\[
y_{k+1} - z_{k+1} = \gamma_k(x_{k+1} - x_k).
\]  \hspace{1cm} (A.1.2)

By the smoothness of \(l(x)\) and the convexity of both \(l(x)\), \(r(x)\), along with update step 5,

\[
F(y_{k+1}) = l(y_{k+1}) + r(y_{k+1}) \leq l(z_{k+1}) + \langle l'(z_{k+1}), y_{k+1} - z_{k+1} \rangle
\]

\[
+ \frac{L}{2}||y_{k+1} - z_{k+1}||^2 + (1 - \gamma_k)r(y_k) + \gamma_kr(x_{k+1})
\]

\[
\leq ((1 - \gamma_k) + \gamma_k)l(z_{k+1}) + \langle l'(z_{k+1}), (1 - \gamma_k)y_k + \gamma_kx_{k+1} - z_{k+1} \rangle
\]

\[
+ \frac{L}{2}||y_{k+1} - z_{k+1}||^2 + (1 - \gamma_k)r(y_k) + \gamma_kr(x_{k+1})
\]

\[
\leq (1 - \gamma_k)(l(y_k) + r(y_k)) + \gamma_k(l(z_{k+1}) + \langle l'(z_{k+1}), x_{k+1} - z_{k+1} \rangle + r(x_{k+1}))
\]

\[
+ \frac{L}{2}||y_{k+1} - z_{k+1}||^2.
\]  \hspace{1cm} (A.1.3)
Substituting equation \[ A.1.2 \] and reorganize,

\[
F(y_{k+1}) \leq (1 - \gamma_k)F(y_k) + \gamma_k(l(z_{k+1}) + \langle x_{k+1} - z_{k+1}, \nabla l(z_{k+1}) \rangle + r(x_{k+1})) + \frac{L\gamma_k^2}{2}||x_{k+1} - x_k||^2
\]

\[
= (1 - \gamma_k)F(y_k) + \gamma_k(l(z_{k+1}) + \langle l'(z_{k+1}), x_{k+1} - z_{k+1} \rangle + r(x_{k+1})) + \frac{\beta_k\gamma_k}{2}||x_{k+1} - x_k||^2
- \frac{\gamma_k}{2}(\beta_k - L\gamma_k)||x_{k+1} - x_k||^2
\]

\[
\leq (1 - \gamma_k)F(y_k) + \gamma_k(l(z_{k+1}) + \langle l'(z_{k+1}), x_{k+1} - z_{k+1} \rangle + r(x_{k+1})) + \frac{\beta_k\gamma_k}{2}||x_{k+1} - x_k||^2.
\]

(A.1.4)

Next, by using equation \[ A.1.1 \] to the last term, it gives

\[
F(y_{k+1}) \leq (1 - \gamma_k)F(y_k) + \gamma_k(l(z_{k+1}) + \langle l'(z_{k+1}), x_{k+1} - z_{k+1} \rangle + r(x_{k+1}))
+ \frac{\beta_k\gamma_k}{2} \left[ ||x_k - x||^2 - ||x_{k+1} - x||^2 + 2\langle x_{k+1} - x, x_{k+1} - x \rangle \right].
\]

(A.1.5)

In the following, we further bound \( \gamma_k\beta_k\langle x_{k+1} - x_k, x_{k+1} - x \rangle \) by the stopping criteria of the subroutine. Namely, by the maximality of \( G_t(v_t) \), for any \( x \), we have

\[
r(x_{k+1}) - r(x) - \langle \phi'(x_{k+1}), x - x_{k+1} \rangle \leq r(x_{k+1}) - r(v_t) - \langle \phi'(x_k), v_t - x_{k+1} \rangle \leq \eta_k.
\]

(A.1.6)

Recall \( \phi'(x_{k+1}) = l'(z_{k+1}) + \beta_k(x_{k+1} - x_k) \), thus combined with the above inequality,

\[
\beta_k\langle x_{k+1} - x_k, x_{k+1} - x \rangle \leq \eta_k + r(x) - r(x_{k+1}) + \langle l'(z_{k+1}), x - x_{k+1} \rangle.
\]

(A.1.7)

Then combine the above inequality with inequality \[ A.1.5 \]

\[
F(y_{k+1}) \leq (1 - \gamma_k)F(y_k) + \gamma_k(l(z_{k+1}) + \langle l'(z_{k+1}), x - z_{k+1} \rangle + r(x)) + \gamma_k\eta_k
+ \frac{\beta_k\gamma_k}{2} \left[ ||x_k - x||^2 - ||x_{k+1} - x||^2 \right]
\]

\[
\leq (1 - \gamma_k)F(y_k) + \gamma_kF(x) + \gamma_k\eta_k + \frac{\beta_k\gamma_k}{2} \left[ ||x_k - x||^2 - ||x_{k+1} - x||^2 \right].
\]

(A.1.8)

Subtract \( F(x) \) from both sides,

\[
F(y_{k+1}) - F(x) \leq (1 - \gamma_k)(F(y_k) - F(x)) + \gamma_k\eta_k + \frac{\beta_k\gamma_k}{2} \left[ ||x_k - x||^2 - ||x_{k+1} - x||^2 \right].
\]

(A.1.9)
By the above recursive relationship, denote $\Gamma_k = \Pi_{i=1}^k (1 - \gamma_i)$ (let $\Gamma_0 = 1$), from $k = 0$ to $K - 1$,

\[
F(y_K) - F(x) \leq \Gamma_{K-1} (1 - \gamma_0) (F(y_0) - F(x)) + \sum_{k=0}^{K-1} \frac{\Gamma_{K-1} \gamma_k \eta_k}{\Gamma_k} \tag{A.1.10}
\]

\[
+ \sum_{k=0}^{K-1} \frac{\Gamma_{K-1} \gamma_k \beta_k}{2 \Gamma_k} \left( ||x_k - x||^2 - ||x_{k+1} - x||^2 \right)
\]

With the sequences $\beta_k = \frac{2L}{k+1}$, $\gamma_k = \frac{2}{k+2}$, $\eta_k = \frac{2LD_0}{K(k+1)}$, we have

\[
1 - \gamma_0 = 0; \quad \Gamma_k = \frac{2}{(k+2)(k+1)}; \quad \Gamma_{K-1} = \frac{2}{K(K+1)}; \quad \frac{\gamma_k \eta_k}{\Gamma_k} = \frac{2LD_0}{K}; \quad \frac{\gamma_k \beta_k}{2 \Gamma_k} = L \tag{A.1.11}
\]

Thus

\[
F(y_K) - F(x) \leq \frac{6LD_0}{K(K+1)} \tag{A.1.12}
\]

For finding an $\epsilon$ solution,

\[
K = \sqrt{\frac{6LD_0}{\epsilon}}. \tag{A.1.13}
\]

### A.1.2 Evaluation bound on LO

Denote $\Phi(v) = \phi(v) + r(v)$. We have

\[
\Phi(u_{t+1}) = \Phi((1 - \alpha_t) u_t + \alpha_t v_t) \leq \Phi((1 - \lambda_t) u_t + \lambda_t v_t)
\]

\[
= \phi(u_t + \lambda_t (v_t - u_t)) + r((1 - \lambda_t) u_t + \lambda_t v_t)
\]

\[
\leq \phi(u_t) + \langle \phi'(u_t), \lambda(v_t - u_t) \rangle + \frac{\beta_t}{2} ||\lambda_t (v_t - u_t)||_2^2 + r(u_t) + \lambda_t (r(v_t) - r(u_t)) \tag{A.1.14}
\]

The first inequality is due to the optimality of $\alpha_t$, where a deterministic $\lambda_t$ takes the place of the varying $\alpha_t$ to ease the following analysis. The second inequality is because $\phi(v)$ is quadratic and $r(v)$ is convex.

By the definition of $\phi(v)$ and plug $G(u_t, v_t) = r(u_t) - r(v_t) - \langle \phi'(u_t), v_t - u_t \rangle$ into the above inequality,

\[
\Phi(u_{t+1}) \leq \Phi(u_t) - \lambda_t G(u_t, v_t) + \frac{\beta \lambda_t^2}{2} ||v_t - u_t||_2^2 \tag{A.1.15}
\]
Subtract optimal value $\Phi(u^*)$ from both sides and denote $\Delta_t = \Phi(u_t) - \Phi(u^*)$.

$$\Delta_{t+1} \leq \Delta_t - \lambda_t G(u_t, v_t) + \frac{\beta \lambda^2}{2} ||v_t - u_t||^2$$  \hspace{1cm} (A.1.16)

$$\leq (1 - \lambda_t)\Delta_t + \frac{\beta \lambda^2}{2} ||v_t - u_t||^2.$$  \hspace{1cm} (A.1.17)

Next, we bound $\Delta_t$. As a duality gap, we have $\Delta_t \leq G(u_t, v_t)$, thus

$$\Delta_{t+1} \leq \Delta_t - \lambda_t G(u_t, v_t) + \frac{\beta \lambda^2}{2} ||v_t - u_t||^2$$  \hspace{1cm} (A.1.18)

By the above recursive relation, consider from iteration 0 to $t - 1$,

$$\Delta_t \leq \Lambda_{t-1}(1 - \lambda_0)\Delta_0 + \frac{\Lambda_{t-1}\beta D_s}{2} \sum_{i=0}^{t-1} \frac{\lambda_i^2}{\Lambda_i} \leq \frac{2\beta D_s}{t + 1}$$  \hspace{1cm} (A.1.19)

Then we move to bound $G(u_t, v_t)$. From inequality (A.1.16), we can also reorganize it as

$$\lambda_t G(u_t, v_t) \leq \Delta_t - \Delta_{t+1} + \frac{\beta \lambda^2}{2} ||v_t - u_t||^2$$  \hspace{1cm} (A.1.20)

Denote $\Lambda_t = \Pi_{i=1}^t (1 - \lambda_t)$ and define $\Lambda_0 = 1$. Divide $\Lambda_t$ from both sides and sum from $i = 0$ to $t$,

$$\sum_{i=0}^t \frac{\lambda_i}{\Lambda_i} G(u_i, v_i) \leq \sum_{i=0}^t \frac{1}{\Lambda_i} (\Delta_i - \Delta_{i+1}) + \sum_{i=0}^t \frac{\lambda_i^2 \beta}{2\Lambda_i} ||v_i - u_i||^2$$  \hspace{1cm} (A.1.21)

Thus,

$$\left(\min_{i=0}^t G(u_i, v_i)\right) \sum_{i=0}^t \frac{\lambda_i}{\Lambda_i} \leq \Delta_0 - \frac{1}{\Lambda_t} \Delta_{t+1} + \sum_{i=1}^t \left(\frac{1}{\Lambda_i} - \frac{1}{\Lambda_{i-1}}\right) \Delta_i + \frac{\beta D_0}{2} \sum_{i=0}^t \frac{\lambda_i^2}{\Lambda_i}$$  \hspace{1cm} (A.1.22)

Substitute $\lambda_i$ and $\Lambda_i$ in, with the bound on $\Delta_i$,

$$\left(\min_{i=0}^t G(u_i, v_i)\right) \frac{(t + 2)(t + 1)}{2} \leq \sum_{i=0}^t (i + 1)\Delta_i + (t + 1)\beta D_s \leq 2(t + 1)\beta D_s + (t + 1)\beta D_s$$  \hspace{1cm} (A.1.23)

Thus,

$$\min_{i=0}^t G(u_i, v_i) \leq \frac{6\beta D_s}{t + 2}$$  \hspace{1cm} (A.1.24)
The inner iteration $m_k$ should satisfy,

$$m_k \geq \frac{6\beta_k D_s}{\eta_k} \quad (A.1.24)$$

By the choice of $\beta_k = \frac{2t}{k+1}$, $\gamma_k = \frac{2}{k+2}$, $\eta_k = \frac{2LD_0}{K(k+1)}$ and setting $m_k = \lceil \frac{6\beta_k D_s}{\eta_k} \rceil$, we have

the total number of LO evaluations

$$T_K = \sum_{k=0}^{K-1} m_k \leq \sum_{k=0}^{K-1} \left( \frac{6D_s K}{D_0} + 1 \right) = \frac{6D_s}{D_0} K^2 + K. \quad (A.1.25)$$

For finding an $\epsilon$ solution,

$$T_K = \frac{36LD_s}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}. \quad (A.1.26)$$

### A.2 Proof of Theorem 3.3.3

#### A.2.1 Outer Loop Analysis

With $x_{k+1} = \bar{u}_m$ and the choice of $m$, for any $x$, we still have the following essential relation

$$r(x_{k+1}) - r(x) - \langle \phi'(x_{k+1}), x - x_{k+1} \rangle \leq \eta_k, \quad (A.2.27)$$

which we will show in the next section. Thus the analysis for outer loop is basically the same as that for the general case. Namely, we have

$$F(y_K) - F(x) \leq \frac{6LD_0}{K(K+1)}. \quad (A.2.28)$$

For finding an $\epsilon$ solution,

$$K = \sqrt{\frac{6LD_0}{\epsilon}}. \quad (A.2.29)$$

#### A.2.2 Inner Loop Analysis

We begin with recalling the following sequences,

$$\alpha_t = \frac{2}{t+2}; \quad \Lambda_t = \frac{2}{(t+2)(t+1)}; \quad \gamma_t = \frac{2}{m(m+1)(t+1)}, \quad (A.2.30)$$
Where $\alpha_t$ is the step weight, $\gamma_t$ is the non-uniform weight for averaging $u_t$. Note that the following relationship holds,

$$\Lambda_{t+1} = \Lambda_t(1 - \alpha_{t+1}); \quad \gamma_t = \frac{2}{m(m+1)} \alpha_t$$  \hspace{1cm} (A.2.31)

Denote $\hat{F}_t = \phi(u_t) + h(\rho_t)$, we recover $G(u_t, x)$ from update scheme. First, by the assumption of Improve and the quadratic of $\phi(u_t)$,

$$\hat{F}_{t+1} = \phi(u_{t+1}) + h(\rho_{t+1})$$

$$\leq \phi(u_t) + \langle \phi'(u_t), \theta_t a_t - \alpha_t u_t \rangle + \frac{\beta}{2} ||\theta_t a_t - \alpha_t u_t||^2 + (1 - \alpha_t)h(\rho_t) + \alpha_t h(\theta_t/\eta_t)$$  \hspace{1cm} (A.2.32)

Next, by the optimality of $\theta_t$, once replaced by $\alpha_t\kappa(x)$, for any $x$, the above relation further gives,

$$\hat{F}_{t+1} \leq \hat{F}_t + \alpha_t \langle \phi'(u_t), \kappa(x)a_t - u_t \rangle + \frac{\beta \alpha_t^2}{2} \cdot \kappa(x)a_t - u_t \rangle^2 - \alpha_t h(\rho_t) + \alpha_t h(\kappa(x))$$  \hspace{1cm} (A.2.33)

Note that $-\alpha_t h(\rho_t) \leq -\alpha_t h(\kappa(u_t))$ and $\langle \phi'(u_t), a_t \rangle \leq \epsilon_t + \min_{a \in A} \langle \phi'(u_t), a \rangle \leq \epsilon_t + \langle \phi'(u_t), e_x \rangle$, where $e_x = x/\kappa(x)$,

$$\hat{F}_{t+1} \leq \hat{F}_t + \alpha_t \left( \langle \phi'(u_t), x - u_t \rangle + h(\kappa(x)) - h(\kappa(u_t)) \right) + \frac{\beta \alpha_t^2}{2} \cdot \kappa(x)a_t - u_t \rangle^2 + \alpha_t \epsilon_t \kappa(x).$$  \hspace{1cm} (A.2.34)

Thus

$$\alpha_t G(u_t, x) \leq \hat{F}_t - \hat{F}_{t+1} + \frac{\beta \alpha_t^2}{2} \cdot \kappa(x)a_t - u_t \rangle^2 + \alpha_t \epsilon_t \kappa(x)$$  \hspace{1cm} (A.2.35)

Denote $\hat{F}_s$ the optimal functional of $\hat{F}$ and $\hat{\Delta}_t = \hat{F}_t - \hat{F}_s$, the above inequality becomes,

$$G(u_t, x) \leq \frac{1}{\alpha_t} (\hat{\Delta}_t - \hat{\Delta}_{t+1}) + \frac{\beta \alpha_t^2}{2} \cdot \kappa(x)a_t - u_t \rangle^2 + \epsilon_t \kappa(x)$$  \hspace{1cm} (A.2.36)

Note that $G(u_t, x)$ is convex in $u_t$,

$$G(u_t, x) = r(u_t) - r(x) - \langle \phi'(u_t), x - u_t \rangle$$

$$= r(u_t) - r(x) + \langle g + \beta(u_t - u), u_t - x \rangle$$

$$= r(u_t) + \beta ||u_t||^2 + \langle g - \beta(u + x), u_t - \rangle [r(x) + \langle g - \beta u, x \rangle].$$  \hspace{1cm} (A.2.37)
Recall the non-uniform weight for each \( u_t \) that \( \gamma_t = \frac{2}{m(m+1)}(t+1), \forall t = 0, 1, ..., m-1. \)

\[
\bar{u}_m = \sum_{t=0}^{m-1} \gamma_t u_t. \tag{A.2.38}
\]

For any \( x \), by the convexity of \( G(u_t, x) \) in \( u_t \), inequality \( A.2.36 \) and the relationship of \( \alpha_t, \gamma_t \) in \( A.2.31 \),

\[
G(\bar{u}_m, x) \leq \sum_{t=0}^{m-1} \gamma_t G(u_t, x) \leq \sum_{t=0}^{m-1} \frac{\gamma_t (\tilde{\Delta}_t - \tilde{\Delta}_{t+1})}{\alpha_t} + \frac{\beta \alpha_t \gamma_t}{2} \| \kappa(x) a_t - u_t \|^2 + \gamma_t \epsilon_t \kappa(x) \\
= \frac{2}{m(m+1)} \left\{ \left[ \sum_{t=0}^{m-1} \frac{1}{\alpha_t} (\tilde{\Delta}_t - \tilde{\Delta}_{t+1}) \right] + \left[ \sum_{t=0}^{m-1} \frac{\beta \alpha_t^2}{2 \alpha_t} \| \kappa(x) a_t - u_t \|^2 \right] + \left[ \sum_{t=0}^{m-1} \frac{\alpha_t}{\alpha_t} \epsilon_t \kappa(x) \right] \right\} 
\]

\[
\leq \sum_{t=0}^{m-1} \left( t + 1 \right) \tilde{\Delta}_t. \tag{A.2.39}
\]

Denote each term in square blankets above as \( T_1, T_2, T_3 \). For \( T_1 \),

\[
T_1 = \tilde{\Delta}_0 - \frac{1}{\lambda_{m-1}} \tilde{\Delta}_m + \sum_{t=1}^{m-1} \left( \frac{1}{\lambda_t} - \frac{1}{\lambda_{t-1}} \right) \tilde{\Delta}_t \n\]

\[
\leq \sum_{t=0}^{m-1} (t + 1) \tilde{\Delta}_t. \tag{A.2.40}
\]

The bound on \( \tilde{\Delta}_t \) is due to the convergence analysis of Theorem 12 and corollary 13 in GCG, which we summarize in the following inequality,

\[
\tilde{\Delta}_t \leq \frac{2(\kappa(x) \delta \beta + \beta D_s)}{t + 3}, \tag{A.2.41}
\]

where \( \delta \) satisfy \( \epsilon_t \leq \frac{\delta \beta \alpha_t}{2} \) hold and \( D_s \) is the upper bound of \( \| \kappa(x) a_t - u_t \|^2 \), both of which are constants. Plug it in, we have

\[
T_1 \leq \sum_{t=0}^{m-1} (t + 1) \frac{2(\kappa(x) \delta \beta + \beta D_s)}{t + 3} \leq 2m \beta (\kappa(x) \delta + D_s). \tag{A.2.42}
\]

Next we consider \( T_2 \),

\[
T_2 \leq \frac{\beta D_s}{2} \sum_{t=0}^{m-1} \frac{\alpha_t^2}{\lambda_t} = \frac{\beta D_s}{2} \sum_{t=0}^{m-1} \frac{2(t + 1)}{t + 2} \leq m \beta D_s \tag{A.2.43}
\]
For $T_3$, with $\epsilon_t \leq \frac{\delta \beta_0}{2}$,

$$T_3 \leq \sum_{t=0}^{m-1} \frac{\delta \beta \kappa(x) \alpha_t^2}{2} \leq m \delta \beta \kappa(x)$$  \hspace{1cm} (A.2.44)

Thus, we have the bound on $G(\bar{u}_t, x)$

$$G(\bar{u}_m, x) \leq \frac{2}{m(m+1)} m \beta \left[ 2(\kappa(x)\delta + D_s) + D_s + \delta \kappa(x) \right]$$

$$\leq \frac{2 \beta \left[ 2(\kappa(x)\delta + D_s) + D_s + \delta \kappa(x) \right]}{m} \hspace{1cm} (A.2.45)$$

For each outer stage $k$, with $\beta_k = \frac{2L}{k+1}$ and $m = \left\lceil \frac{2K(2(\kappa(x)\delta + D_s) + D_s + \delta \kappa(x))}{D_0} \right\rceil$, we have

$$G_{k+1} = G(\bar{u}_m, x) \leq \eta_k = \frac{2LD_0}{K(k+1)}$$  \hspace{1cm} (A.2.46)

For convex problem, the total number of LO evaluation is

$$\sum_{k=0}^{K-1} m_k \leq \frac{2K^2(2(\kappa(x)\delta + D_s) + D_s + \delta \kappa(x))}{D_0} + K. \hspace{1cm} (A.2.47)$$

Substitute $K$ in, we can find an $\epsilon$ solution by

$$T_K \leq \frac{12L(2(\kappa(x)\delta + D_s) + D_s + \delta \kappa(x))}{\epsilon} + \sqrt{\frac{6LD_0}{\epsilon}}$$  \hspace{1cm} (A.2.48)
Appendix B

Proofs of Materials in Chapter 4

B.1 Proof of Theorem 4.3.2 and Theorem 4.3.3

The proof of Theorem 4.3.2 and Theorem 4.3.3 is a combination of the proof in 26 and 146. We proceed the proof by two steps. First we prove that for the PA approximation \( \hat{f}(x) = l(x) + \hat{r}(x) \) and its global optimal value \( \hat{f}(\hat{x}^*) \), \( \hat{f}(x^t) - \hat{f}(\hat{x}^*) \) converges linearly in expectation. Then we conclude the proof using Lemma 2.2.2 which tells that the surrogate \( \hat{f}(x) \) and the original \( F(x) \) can be arbitrarily close.

For the first step, we use the following Lyapunov function,

\[
T^t = Q^t + \left( c_1 + \frac{c_2}{\eta} \right) ||x^t - \hat{x}^*||^2_2 + c_2(\hat{f}(x^t) - \hat{f}(\hat{x}^*)) , \tag{B.1.1}
\]

\[
Q^t = \frac{1}{n} \sum_{i=1}^{n} l_i(\phi_i^t) - l(\hat{x}^*) - \frac{1}{n} \sum_{i=1}^{n} \langle \nabla l_i(\hat{x}^*), \phi_i^t - \hat{x}^* \rangle , \tag{B.1.2}
\]

where \( \eta \) is the step size and \( c_1, c_2 \) are constants to be specified in the following proof. Since the step size \( \eta \) is related to \( \epsilon \), our choice of parameters should be different from 26, otherwise, the difference between \( \hat{f}(x) \) and \( F(x) \) cannot be small enough. We borrow techniques and intermediate results from 26, which could be summarized into the following three lemmas. We omit the proof of these lemmas as they are paraphrase of those can be found in 26. In the following, the expectation is conditioned on information up to iteration \( t \), where \( i^t \) is the randomly picked index, thus is a random variable.
Lemma B.1.1. ([26], Theorem 1) 
\( \mathbb{E}[Q^{t+1}] \) has the following iterative relationship with \( Q^t \),

\[
\mathbb{E}[Q^{t+1}] = (1 - \frac{1}{n})Q^t + \frac{1}{n} \left[ l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle \right].
\] (B.1.3)

The next lemma bounds \( \mathbb{E}c_1||x^{t+1} - \hat{x}^*||_2^2 \). Recall that \( \hat{x}^* \) is the optimal point of the approximate function \( \hat{f}(x) \), we have:

Lemma B.1.2. ([26], Theorem 1)

\[
c_1\mathbb{E}||x^{t+1} - \hat{x}^*||_2^2 \leq (1 - \eta\mu)c_1||x^t - \hat{x}^*||_2^2 + 2(1 + \beta^{-1})c_1\eta^2LQ^t
\]

\[
+ ((1 + \beta)c_1\eta^2 - \frac{c_1\eta}{L})\mathbb{E}[||\nabla l_{i^*}(x^t) - \nabla l_{i^*}(\hat{x}^*)||_2^2]
\]

\[
- \left( 2c_1\eta^2\beta\mu + 2c_1\eta(1 - \frac{\mu}{L}) \right) \left[ l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle \right].
\] (B.1.4)

Finally we prepare ourselves with the lemma bounding \( \mathbb{E} \left[ c_2(\hat{f}(x^{t+1}) - \hat{f}(\hat{x}^*)) + \frac{c_2}{\eta}||x^{t+1} - \hat{x}^*||_2^2 \right] \).

Lemma B.1.3. ([26], Theorem 2) For some \( \beta > 0 \),

\[
c_2\mathbb{E}\left[ \hat{f}(x^{t+1}) - \hat{f}(\hat{x}^*) \right] + \frac{c_2}{2\eta}\mathbb{E}||x^{t+1} - \hat{x}^*||_2^2 \leq \frac{c_2}{2\eta}||x^t - \hat{x}^*||_2^2
\]

\[
+ 2(1 + \beta^{-1})c_2\eta LQ^t + (1 + \beta)c_2\eta\mathbb{E}[||\nabla l_{i^*}(x^t) - \nabla l_{i^*}(\hat{x}^*)||_2^2].
\] (B.1.5)

B.1.1 Proof of Theorem 4.3.2 General Convex Loss Function Case

With Lemma 1-3, we have:

\[
\mathbb{E}[T^{t+1}] = \mathbb{E} \left[ Q^{t+1} + (c_1 + \frac{c_2}{2\eta})||x^{t+1} - \hat{x}^*||_2^2 + c_2[\hat{f}(x^{t+1}) - \hat{f}(\hat{x}^*)) \right]
\]

\[
\leq \left[ Q^t + (c_1 + \frac{c_2}{2\eta})||x^t - \hat{x}^*||_2^2 \right]
\]

\[
+ \left( 2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2L - \frac{1}{n} \right)Q^t
\]

\[
+ \left( (1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1\eta}{L} \right)\mathbb{E}[||\nabla l_{i^*}(x^t) - \nabla l_{i^*}(\hat{x}^*)||_2^2]
\]

\[
+ \left( \frac{1}{n} - 2c_1\eta \right) \left[ l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle \right].
\] (B.1.6)
With the choice of parameters $c_1, c_2, \frac{1}{\kappa}, \beta$ and $\eta$, the terms in big round brackets are non-positive (we defer this discussion to the end of the proof). We leave out these non-positive terms,

$$
\mathbb{E}\left[ Q^{t+1} + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^{t+1} - \hat{x}^*||_2^2 + c_2[\hat{f}(x^{t+1}) - \hat{f}(\hat{x}^*)] \right] \leq \left[ Q^t + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^t - \hat{x}^*||_2^2 \right].
$$

(B.1.7)

By this iteration relationship and considering from step 0 to $t$, we have

$$
\mathbb{E}c_2\left[ \hat{f}(x^t) - \hat{f}(\hat{x}^*) \right] \leq \frac{1}{t} \mathbb{E}\left[ c_2 \sum_{i=0}^{t} [f(x^i) - \hat{f}(\hat{x}^*)] \right]
$$

$$
\leq \frac{1}{t} \left( Q^0 + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^0 - \hat{x}^*||_2^2 - \mathbb{E}\left[ Q^t + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^t - \hat{x}^*||_2^2 \right] \right)
$$

$$
\leq \frac{1}{t} \left( Q^0 + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^0 - \hat{x}^*||_2^2 \right),
$$

(B.1.8)

where $Q^0 = \frac{1}{n} \sum_{i=1}^{n} l_i(\phi_i^0) - l(\tilde{x}^*) - \frac{1}{n} \sum_{i=1}^{n} \langle \nabla l_i(\tilde{x}^*), \phi_i^0 - \hat{x}^* \rangle.$

(B.1.9)

Hence as long as $t \geq \frac{1}{c_2\epsilon} \left( Q^0 + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^0 - \hat{x}^*||_2^2 \right)$, we get:

$$
\mathbb{E}\left[ \hat{f}(\tilde{x}^t) - \hat{f}(\hat{x}^*) \right] \leq \epsilon.
$$

(B.1.10)

We have i) $\mathbb{E}[F(\tilde{x}^t) - \hat{f}(\tilde{x}^t)] \leq \epsilon$ by Lemma 1 (i.e. $r(x) - \hat{r}(x) \leq \epsilon$, $\forall x$); ii) $\mathbb{E}[\hat{f}(\tilde{x}^t) - \hat{f}(\hat{x}^*)] \leq \epsilon$ by inequality [B.1.10]; iii) $\mathbb{E}[\hat{f}(\hat{x}^*) - F(\hat{x}^*)] \leq 0$ by Lemma 1 (i.e. $0 \leq r(x) - \hat{r}(x), \forall x$). By i)-iii), we have

$$
\mathbb{E}[F(\tilde{x}^t) - F(\hat{x}^*)] = \mathbb{E}[\left( (F(\tilde{x}^t) - \hat{f}(\tilde{x}^t)) + (\hat{f}(\tilde{x}^t) - \hat{f}(\hat{x}^*)) + (\hat{f}(\hat{x}^*) - F(\hat{x}^*)) \right)]
$$

$$
\leq \epsilon + \epsilon + 0.
$$

(B.1.11)

Thus, as long as $t \geq \frac{1}{c_2\epsilon} \left( Q^0 + \left( c_1 + \frac{c_2}{2\eta}\right) ||x^0 - \hat{x}^*||_2^2 \right)$, $\mathbb{E}[F(\tilde{x}^t) - F(\hat{x}^*)] \leq 2\epsilon$.

**Verification of non-positiveness for general convex case in eq. (B.1.6):**

In the following, we show the non-positiveness of the terms in the round bracket in eq. (B.1.6). For the four inequalities, when $\eta < \min\left( \frac{1}{2L}, \frac{2\epsilon}{M^2} \right)$, $c_1 = \frac{1}{2\eta}L$, $c_2 = \frac{1}{2\eta}$,
\[
\frac{1}{2n}\left(\frac{1}{2nL\beta} - 1\right), \beta = 1, \text{ we need to ensure,}
\]
\[
I_1 : 2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2 L - \frac{1}{n} \leq 0, \quad (B.1.12)
\]
\[
I_2 : (1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1\eta}{L} \leq 0, \quad (B.1.13)
\]
\[
I_3 : \frac{1}{n} - 2c_1\eta \leq 0. \quad (B.1.14)
\]

\(I_3\) is 0 when \(c_1 = \frac{1}{2m};\ c_2 = \frac{1}{2n}\left(\frac{1}{2nL\beta} - 1\right) = c_1\eta\left(\frac{1}{2nL\beta} - 1\right) \leq c_1\eta\left(\frac{1}{\eta L(1+\beta)} - 1\right), \) thus \(I_2\) is satisfied. Substituting \(c_1\) and \(c_2\) into \(I_1,\) it gives \((1 + \frac{1}{\beta})\frac{1}{\beta} \leq 1,\) meanwhile, \(2nL\frac{1}{\beta} \leq 1,\) both of which are satisfied when \(\beta = 1\) under \(\eta < \frac{1}{2L}.\)

### B.1.2 Proof of Theorem 4.3.3 Strongly Convex Loss Function Case

Combining Lemma 1-3 and rearranging the term, we can get:

\[
\mathbb{E}[T^{t+1}] = \mathbb{E}\left[\hat{Q}^{t+1} + (c_1 + \frac{c_2}{\eta})||\hat{x}^{t+1} - \hat{x}^*||^2_2 + c_2[\hat{f}(x^{t+1}) - \hat{f}(\hat{x}^*)]\right]
\]
\[
\leq \left[\hat{Q}^t + (c_1 + \frac{c_2}{2\eta})||x^t - \hat{x}^*||^2_2\right]
\]
\[
+ \left(2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2 L - \frac{1}{n}\right)Q^t - \eta\mu c_1||x^t - \hat{x}^*||^2_2
\]
\[
+ \left((1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1\eta}{L}\right)\mathbb{E}[||\nabla l(x^t) - \nabla l(\hat{x}^*)||^2_2]
\]
\[
+ \left(\frac{1}{n} - 2c_1\eta^2\beta\mu - 2c_1\eta(1 - \frac{\mu}{L})\right)\left[l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle\right].
\]

(B.1.15)

Adding a nonnegative term \(c_2\left(1 - \frac{1}{\kappa}\right)[\hat{f}(x^t) - \hat{f}(\hat{x}^*)]\) to the RHS with \(\frac{1}{\kappa} \in (0,1),\) we have:

\[
\mathbb{E}[T^{t+1}] \leq T^t - \frac{c_2}{\kappa}[\hat{f}(x^t) - \hat{f}(\hat{x}^*)]
\]
\[
+ \left(2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2 L - \frac{1}{n}\right)Q^t - \eta\mu c_1||x^t - \hat{x}^*||^2_2
\]
\[
+ \left((1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1\eta}{L}\right)\mathbb{E}[||\nabla l(x^t) - \nabla l(\hat{x}^*)||^2_2]
\]
\[
+ \left(\frac{1}{n} - 2c_1\eta^2\beta\mu - 2c_1\eta(1 - \frac{\mu}{L})\right)\left[l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle\right].
\]

(B.1.16)
After further extracting a $-\frac{1}{\kappa}T^t$ term from the RHS, we have

$$
\mathbb{E}[T^{t+1}] - T^t \leq -\frac{1}{\kappa}T^t + \left(\frac{1}{\kappa} + 2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2 L - \frac{1}{n}\right)Q^t \\
+ \left(\frac{1}{\kappa}(c_1 + \frac{c_2}{2\eta}) - \eta \mu c_1\right)||x^t - \hat{x}^*||_2^2 \\
+ \left((1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1\eta}{L}\right)\mathbb{E}[||\nabla l_i(x^t) - \nabla l_i(\hat{x}^*)||_2^2] \\
+ \left(\frac{1}{n} - 2c_1\eta^2 \beta \mu - 2c_1\eta(1 - \frac{\mu}{L})\right)\left[l(x^t) - l(\hat{x}^*) - \langle \nabla l(\hat{x}^*), x^t - \hat{x}^* \rangle\right].
$$

(B.1.17)

With the choice of parameters $c_1, c_2, \frac{1}{\kappa}, \beta$ and $\eta$, the terms in big round brackets are non-positive (we defer this discussion to the end of the proof). We leave out these non-positive terms and by iteration relation,

$$
\mathbb{E}[\hat{f}(x^t) - \hat{f}(\hat{x}^*)] \leq \mathbb{E}[T^t] \leq (1 - \frac{1}{\kappa})^t T^0 \\
= (1 - \frac{1}{\kappa})^t \left[Q^0 + (c_1 + \frac{c_2}{\eta})||x^0 - \hat{x}^*||_2^2 + c_2(\hat{f}(x^0) - \hat{f}(\hat{x}^*))\right],
$$

(B.1.18)

where

$$
Q^0 = \frac{1}{n} \sum_{i=1}^{n} l_i(\phi_i^0) - l(\hat{x}^*) - \frac{1}{n} \sum_{i=1}^{n} \langle \nabla l_i(\hat{x}^*), \phi_i^0 - \hat{x}^* \rangle.
$$

(B.1.19)

Hence, as long as $t \geq \log \frac{T^0}{\epsilon} / \log(1 - \frac{1}{\kappa})$, we have:

$$
\mathbb{E}[\hat{f}(x^t) - \hat{f}(\hat{x}^*)] \leq \epsilon.
$$

(B.1.20)

Similar to the reasoning in the proof of Theorem 4.3.2 and by Lemma 2.2.2,

$$
\mathbb{E}[F(x^t) - F(\hat{x}^*)] = \mathbb{E}[(F(x^t) - \hat{f}(x^t)) + (\hat{f}(x^t) - \hat{f}(\hat{x}^*)) + (\hat{f}(\hat{x}^*) - F(\hat{x}^*))] \\
\leq \epsilon + \epsilon + 0.
$$

(B.1.21)

Finally, we conclude that, as long as $t \geq \log \frac{T^0}{\epsilon} / \log(1 - \frac{1}{\kappa})$, we have

$$
\mathbb{E}[F(x^t) - F(\hat{x}^*)] \leq 2\epsilon.
$$

Verification of non-positiveness for strongly convex case in eq. (B.1.17):
In the following, we show the non-positiveness of the terms in the round bracket in eq. (B.1.17). For the four inequalities, when \( \eta < \min\left(\frac{1}{2L}, \frac{2\mu}{M^2}\right) \), \( c_1 = \frac{1}{2\eta} \frac{L}{L-\mu} \), \( c_2 = c_1 \eta \left(\frac{1}{2\eta L \beta} - 1\right) \), \( \frac{1}{\kappa} = \frac{2\eta\mu}{1 + \frac{2\eta\mu}{L}} \), we shall verify

\[
I_1 : \frac{1}{\kappa} + 2(1 + \beta^{-1})(c_1 + \frac{c_2}{\eta})\eta^2 L - \frac{1}{n} \leq 0, \quad \text{B.1.22}
\]

\[
I_2 : \frac{1}{\kappa} \left(c_1 + \frac{c_2}{\eta}\right) - \eta \mu c_1 \leq 0, \quad \text{B.1.23}
\]

\[
I_3 : (1 + \beta)(c_1 + \frac{c_2}{\eta})\eta^2 - \frac{c_1 \eta}{L} \leq 0, \quad \text{B.1.24}
\]

\[
I_4 : \frac{1}{n} - 2c_1 \eta^2 \beta \mu - 2c_1 \eta \left(1 - \frac{\mu}{L}\right) \leq 0. \quad \text{B.1.25}
\]

First consider \( I_4 \), when \( c_1 = \frac{1}{2\eta} \frac{L}{L-\mu} \),

\[
c_1 = \frac{1}{2\eta} \frac{L}{L-\mu} > \frac{1}{2\eta} \frac{1}{1 - \frac{\mu}{L}} \frac{1}{1 + \mu \eta \beta}. \quad \text{B.1.26}
\]

Thus, \( I_4 \) is satisfied:

\[
\frac{1}{n} - 2c_1 \eta^2 \beta \mu - 2c_1 \eta \left(1 - \frac{\mu}{L}\right) \leq 0. \quad \text{B.1.27}
\]

Next for \( I_3 \), under \( c_2 = \frac{1}{2\eta} \frac{L}{L-\mu} \left(\frac{1}{2\eta L \beta} - 1\right) \) and the fact \( \beta > 1 \),

\[
c_2 = c_1 \eta \left(\frac{1}{2\eta L \beta} - 1\right) \leq c_1 \eta \left(\frac{1}{(1 + \beta)\eta L} - 1\right), \quad \text{B.1.28}
\]

which is equal to \( I_3 \), thus we have verified \( I_3 \). In addition, we need \( 2\eta L \beta < 1 \) to guarantee \( c_2 > 0 \) which is satisfied when substituting \( \beta \) in.

Now we move to \( I_2 \), as \( \frac{1}{\beta} < 1 \) and \( \frac{1}{\kappa} = \frac{2\eta \mu}{1 + \frac{2\eta \mu}{L}} \),

\[
\frac{1}{\kappa} = \frac{2\eta \mu}{1 + \frac{2\eta \mu}{L}} < \frac{2\eta \mu}{1 + \frac{1}{L} \frac{1}{\beta}}, \quad \text{B.1.29}
\]

which is equivalent to \( I_2 \) after substituting \( c_1, c_2 \) into \( I_2 \). Apparently, as \( 2\eta L \beta < 1 \), \( 2\eta \mu \leq 2\eta L \leq 2\eta L \beta < 1 \) and \( 1 + \frac{1}{2\eta L} > 1 + \frac{1}{2\eta L \beta} > 2 \), thus \( \frac{1}{\kappa} < 1 \) is also satisfied.

Finally, we deal with \( I_1 \). When \( \eta < \frac{1}{2L} \), we have \( \eta \leq \frac{1}{2L} + \frac{1}{4L^2} \frac{2\eta \mu^2 (L - \mu)}{L} \), given \( \eta < \frac{1}{2\eta \mu} \), which is equivalent to

\[
4L^2 \eta^2 + \left(2L + \frac{2\eta \mu^2 (L - \mu)}{L}\right) \eta \leq 0 \leq 2\left(1 - \frac{\mu}{L}\right). \quad \text{B.1.30}
\]

Rearranging the terms, we get

\[
(1 + 2\eta L)2\eta L \leq \left(1 - 2\eta \mu \right)2\left(1 - \frac{\mu}{L}\right). \quad \text{B.1.31}
\]
Note the fact that \((1 - 2\eta\mu)2(1 - \frac{\mu}{L}) < \left(1 - \frac{2\eta\mu2\eta L}{2\eta L + 1}\right)2(1 - \frac{\mu}{L}) < 2\). We have \((1 + 2\eta L)2\eta L\) is strictly less than \(\left(1 - \frac{2\eta\mu2\eta L}{2\eta L + 1}\right)2(1 - \frac{\mu}{L})\) under \(\eta < \frac{1}{2L}\). After we substituting \(\frac{1}{\kappa}, c_1, c_2\) into \(I_1\),

\[
(1 + \frac{1}{\beta}) \frac{1}{\beta} \leq \left(1 - \frac{2\eta\mu2\eta L}{2\eta L + 1}\right)2(1 - \frac{\mu}{L}), \tag{B.1.32}
\]

with the additional requirement for \(\beta\) that \(2\eta L < \frac{1}{\beta} \leq 1\), we can guarantee that there exists some \(\beta\) (e.g. taking average) to satisfy the relationship,

\[
(1 + 2\eta L)2\eta L < \left(1 + \frac{1}{\beta}\right) \frac{1}{\beta} \leq \left(1 - \frac{2\eta\mu2\eta L}{2\eta L + 1}\right)2(1 - \frac{\mu}{L}). \tag{B.1.33}
\]
Appendix C

Proofs of Materials in Chapter 6

C.1 Proof of Materials in Section 6.3

C.1.1 Proof of Proposition 6.3.1

Proof. Define the indicator function of set $C$ as follows,

$$
X_C(A) = \begin{cases} 
0, & A \in C \\
\infty, & A \notin C.
\end{cases}
$$  \hfill (C.1.1)

We prove the Proposition by computing the Fenchel Biconjugate of the indicator function of set $C^{(sp)}_k$. Recall the block diagonal matrix $\hat{A}$. $X_{C^{(sp)}_k}(A)$ can be equivalently interpreted as,

$$
C^{(sp)}_k = \left\{ \hat{A} : \|\hat{A}\|_2 = \|\sigma_{\hat{A}}\|_{\infty} \leq \alpha, \sum_{i=1}^{n_k} rank(\hat{A}^{(i)}) = rank(\hat{A}) = \|\sigma_{\hat{A}}\|_0 \leq k \right\}.
$$  \hfill (C.1.2)

We first compute the Fenchel conjugate of $X_{C^{(sp)}_k}(A)$ by

$$
X_{C^{(sp)}_k}(B) = \sup_A \langle A, B \rangle - X_{C^{(sp)}_k}(A)
$$

$$
\equiv \sup_{\hat{A}} \frac{1}{\alpha} \langle \hat{A}, \hat{B} \rangle - X_{C^{(sp)}_k}(\hat{A})
$$

$$
\equiv \sup_{\sigma_{\hat{A}}} \frac{1}{\alpha} \langle \sigma_{\hat{A}}, \sigma_{\hat{B}} \rangle - X_{C^{(sp)}_k}(\sigma_{\hat{A}})
$$

$$
= \| (\sigma_{\hat{B}})[1:k] \|_1,
$$  \hfill (C.1.3)
where the equality \( (i) \) is by Von Neumann’s trace inequality with \( \hat{B} \) shares the same unitarily matrices \( \hat{U} \) and \( \hat{V} \) with \( \hat{A} \); the last equality is by the property of the vector \( \ell_1 \) norm and eq. (C.1.2) which also amounts to \( C_k^{(\infty)} = \{ \frac{1}{\alpha} \sigma_{\hat{A}} : \| \frac{1}{\alpha} \sigma_{\hat{A}} \|_\infty \leq 1, \| \sigma_{\hat{A}} \|_0 \leq k \} \).

Then, the Fenchel Biconjugate is computed by the Fenchel conjugate of \( X^\ast (C_k^{(sp)}) \) as follows,

\[
X^{**}_{C_k^{(sp)}}(A) = \sup_{A} \langle B, A \rangle - X^\ast_{C_k^{(sp)}}(B)
\]

\[
= \sup_{\sigma_B} \frac{1}{\alpha} \langle \sigma_B, \sigma_{\hat{A}} \rangle - \| (\sigma_B)_{1:k} \|_1
\]

\[
= \begin{cases} 
0, & \| \frac{1}{\alpha} \sigma_{\hat{A}} \|_\infty \leq 1, \frac{1}{k} \| \sigma_{\hat{A}} \|_1 \leq 1 \\
\infty, & \text{otherwise}
\end{cases}
\]

\[
= X_{\text{conv}(C_k^{(sp)})},
\]

which indicates that the convex envelop of the sum of tubal rank on the \( \alpha \)-scaled tensor spectral norm ball is the general tensor nuclear norm. In addition, by substituting \( \alpha = 1 \) or \( n_3 \) into the above proof process, one can immediately recover the relaxation of the two existing specific TNN notion.

\[\square\]

C.1.2 Proof of Proposition 6.3.2

Proof. We prove the Proposition by computing the Fenchel Biconjugate of the indicator function of set \( C_k^{(Fro)} \). \( C_k^{(Fro)} \) can be equivalently interpreted as,

\[
C_k^{(Fro)} = \left\{ \hat{A} : \| \hat{A} \|_F = \| \sigma_{\hat{A}} \|_2 \leq n_3, \sum_{i=1}^{n_3} \text{rank}(\hat{A}^{(i)}) = \text{rank}(\hat{A}) = \| \sigma_{\hat{A}} \|_0 \leq k \right\}.
\]

In particular, we extract the part of the singular values from above (note that \( \sigma_{\hat{A}}/n_3 \) has the same cardinality as \( \sigma_{\hat{A}} \)) and denote it by

\[
C_k^{(sv)} = \left\{ \sigma_{\hat{A}} : \| \sigma_{\hat{A}}/n_3 \|_2 \leq 1, \| \sigma_{\hat{A}}/n_3 \|_0 \leq k \right\},
\]

which amounts to

\[
C_k^{(2)} = \left\{ v \in \mathbb{R}^D : \| v \|_2 \leq 1, \| v \|_0 \leq k \right\}.
\]
With the above equivalence relationships, the Fenchel conjugate of \( \mathcal{X}_{c_k(Fro)}^{\star}(A) \) can be computed by

\[
\mathcal{X}_{c_k(Fro)}^{\star}(B) = \sup_{A} \langle A, B \rangle - \mathcal{X}_{c_k(Fro)}^{\star}(A) \\
= \sup_{\tilde{A}} \frac{1}{n_3} \langle \tilde{A}, \tilde{B} \rangle - \mathcal{X}_{c_k(Fro)}^{\star}(A) \\
\overset{(i)}{=} \sup_{\sigma_{\tilde{A}}} \frac{1}{n_3} \langle \sigma_{\tilde{A}}, \sigma_{\tilde{B}} \rangle - \mathcal{X}_{c_k(Fro)}^{\star}(A) \\
= \sup_{\sigma_{\tilde{A}}/n_3, \sigma_{\tilde{B}}} \sigma_{\tilde{A}} - \mathcal{X}_{c_k^{(sv)}}(\sigma_{\tilde{A}}) \\
= \| (\sigma_{\tilde{B}})[1:k] \|_2 = \| \sigma_{\tilde{B}} \|_{vp,k} = \| \tilde{B} \|_{msp,k},
\]

where the equality (\( i \)) is by Von Neumann’s trace inequality with \( \tilde{A} \) sharing same unitary matrices \( \tilde{U} \) and \( \tilde{V} \) from SVD with \( \tilde{B} \). Then, the Fenchel Biconjugate is computed by the Fenchel conjugate of \( \mathcal{X}_{c_k(Fro)}^{\star}(B) \) as follows,

\[
\mathcal{X}_{c_k(Fro)}^{\star\star}(A) = \sup_{A} \langle B, A \rangle - \mathcal{X}_{c_k(Fro)}^{\star}(B) \\
= \sup_{\sigma_B} \frac{1}{n_3} \langle \sigma_B, \sigma_{\tilde{A}} \rangle - \| \sigma_B \|_{\star}^{\star} \\
= \mathcal{X}_{\frac{1}{n_3} \| \sigma_{\tilde{A}} \|_{vp,k} \leq 1}(\sigma_{\tilde{A}}),
\]

where the last equality is because the Fenchel conjugate of a norm (i.e. dual norm of the \( k \)-support norm) is the indicator function of the unit ball of its dual norm (i.e. the \( k \)-support norm). Also, the constraint \( \| \frac{1}{n_3} \sigma_{\tilde{A}} \|_2 \leq 1 \) (again by applying the property of the \( k \)-support norm to vector \( \frac{1}{n_3} \sigma_{\tilde{A}} \) gives \( \| \frac{1}{n_3} \sigma_{\tilde{A}} \|_2 = \sqrt{n_3} \| A \|_{F} \leq 1 \), which is the scaled tensor Frobenius norm ball of \( \| A \|_{F} \leq \sqrt{n_3} \). As a result, the TSP-\( k \) norm takes the form as

\[
\| A \|_{tsp,k} = \frac{1}{n_3} \| \sigma_{\tilde{A}} \|_{vp,k} = \frac{1}{n_3} \| \tilde{A} \|_{msp,k}.
\]

\[\Box\]
C.1.3 Proof of Proposition 6.3.4

Proof. When $k = 1$, we have $l = 0$, and subsequently we have $\|A\|_{tsp, 1} = \frac{1}{n_3} \|\sigma_A\|_1 = \frac{1}{n_3} \sum_{i=1}^{n_3} \|\sigma_A(i)\|_1 = \|A\|_{ts, \text{avg}}$; When $k = D$, the dual norm becomes

$$\|A\|_{tsp, k} = \|\sigma_A^{(1:D)}\|_2 = \|\sigma_A\|_2 = \|\hat{A}\|_F = \|\hat{A}\|_{msp, k},$$

(C.1.11)

which indicates the primal norm $\|A\|_{tsp, k} = \frac{1}{n_3} \|\hat{A}\|_{msp, k} = \frac{1}{n_3} \|\sigma\|_2 = \frac{1}{n_3} \|\hat{A}\|_F = \frac{1}{\sqrt{n_3}} \|A\|_F$. \hfill \qed

C.2 Proof Materials in Section 6.4

C.2.1 Proof of Proposition 6.4.1

Proof. In order to compute $\mathcal{L}^\#$:

$$\mathcal{L}^\# = \arg \max_{\mathcal{L}} \frac{\beta}{2} (\|\mathcal{L}\|_{tsp, k}^*)^2 + \frac{1}{2} \|\mathcal{L} - \mathcal{J}\|_F^2,$$  \hfill (C.2.12)

we can first convert the problem to Fourier domain via FFT and then recover the result via IFFT. The equivalent problem after FFT is

$$\hat{\mathcal{L}}^\# = \arg \max_{\hat{\mathcal{L}}} \frac{\beta}{2} (\|\hat{\mathcal{L}}\|_{msp, k}^*)^2 + \frac{1}{2n_3} \|\hat{\mathcal{L}} - \hat{\mathcal{J}}\|_F^2,$$ \hfill (C.2.13)

With $\hat{\mathcal{L}}^\#$ sharing the same unitary matrices $\hat{U}$ and $\hat{V}$ with $\hat{\mathcal{J}}$, it suffices to compute the proximal operator of the vector dual $k$-support norm:

$$\sigma_{\hat{\mathcal{L}}}^* = \arg \max_{\sigma_{\hat{\mathcal{L}}}} \frac{\beta n_3}{2} (\|\sigma_{\hat{\mathcal{L}}}\|_{vp, k}^*)^2 + \frac{1}{2} \|\sigma_{\hat{\mathcal{L}}} - \sigma_{\hat{\mathcal{J}}}\|_2^2.$$

(C.2.14) \hfill \qed

C.2.2 Detailed derivation of preconditioned ADMM

To solve eq.(16) in the paper via ADMM-type algorithm, the augmented Lagrangian is given by

$$\mathcal{D}_\rho(\mathcal{L}, \mathcal{E}, \mathcal{J}) = \frac{1}{2} \|\mathcal{L}\|_{tsp, k}^2 + \lambda \|\mathcal{E}\|_1 + \langle \mathcal{J}, \mathcal{M}(\mathcal{X} - \mathcal{L}) - \mathcal{E}\rangle + \frac{\rho}{2} \|\mathcal{M}(\mathcal{X} - \mathcal{L}) - \mathcal{E}\|_F^2.$$

(C.2.15)

We then carry out alternative update to the variables $\mathcal{L}$ and $\mathcal{E}$ at every iteration.
Update of $\mathcal{L}_t$

\[
\mathcal{L}_t = \arg\min_{\mathcal{L}} \frac{1}{2} \| \mathcal{L} \|^2_{tsp,k} + \langle J_{t-1}, \mathcal{M}(\mathcal{X} - \mathcal{L}) - \mathcal{E}_{t-1} \rangle + \frac{\rho}{2} \| \mathcal{M}(\mathcal{X} - \mathcal{L}) - \mathcal{E}_{t-1} \|^2_F. \tag{C.2.16}
\]

To separate the linear map $\mathcal{M}$ apart from $\mathcal{L}$, the preconditioned ADMM approximates $\frac{1}{2} \| \mathcal{M}(\mathcal{X} - \mathcal{L}) - \mathcal{E}_{t-1} \|^2_F$ with second order Taylor expansion around $\mathcal{L}_{t-1}$, as

\[
\frac{1}{2} \| \mathcal{M}(\mathcal{X} - \mathcal{L}_{t-1}) - \mathcal{E}_{t-1} \|^2_F - \langle \mathcal{M}^\top (\mathcal{M}(\mathcal{X} - \mathcal{L}_{t-1}) - \mathcal{E}_{t-1}), \mathcal{L} - \mathcal{L}_{t-1} \rangle + \frac{\eta}{2} \| \mathcal{L} - \mathcal{L}_{t-1} \|^2_F. \tag{C.2.17}
\]

Incorporating eq. (C.2.17) into eq. (C.2.16) gives

\[
\mathcal{L}_t = \arg\min_{\mathcal{L}} \frac{1}{2\rho\eta} \| \mathcal{L} \|^2_{tsp,k} + \frac{1}{2} \| \mathcal{L} - (\mathcal{L}_{t-1} + \frac{1}{\eta} \mathcal{M}^\top (\mathcal{M}(\mathcal{X} - \mathcal{L}_{t-1}) - \mathcal{E}_{t-1}) + \frac{J_{t-1}}{\rho\eta} \|_F^2 \tag{C.2.18}
\]

\[
= \text{Prox}_{\frac{1}{2\rho\eta} \| \cdot \|^2_{tsp,k}} \left( \mathcal{L}_{t-1} + \frac{1}{\eta} \mathcal{M}^\top (\mathcal{M}(\mathcal{X} - \mathcal{L}_{t-1}) - \mathcal{E}_{t-1}) \right)
+ \frac{J_{t-1}}{\rho\eta}
\]

Update of $\mathcal{E}_t$

\[
\mathcal{E}_t = \arg\min_{\mathcal{E}} \lambda \| \mathcal{E} \|_1 + \langle J_{t-1}, \mathcal{M}(\mathcal{X} - \mathcal{L}_t) - \mathcal{E} \rangle + \frac{\rho}{2} \| \mathcal{M}(\mathcal{X} - \mathcal{L}_t) - \mathcal{E} \|^2_F
= \arg\min_{\mathcal{E}} \frac{\lambda}{\rho} \| \mathcal{E} \|_1 + \frac{1}{2} \| \mathcal{E} - (\mathcal{M}(\mathcal{X} - \mathcal{L}_t) + \frac{1}{\rho} J_{t-1}) \|_F
= \text{Prox}_{\frac{\lambda}{\rho} \| \cdot \|_1} \left( \mathcal{M}(\mathcal{X} - \mathcal{L}_t) + \frac{1}{\rho} J_{t-1} \right)
\]

which then follows by the element-wise soft-thresholding operation.

### C.2.3 Computational Complexity Analysis for Algorithm 14

Suppose the input tensor size is $n_1 \times n_2 \times n_3$. The step-by-step computational complexities for Algorithm 14 are as follows:
• Step 2: Element-wise addition/subtraction take $O(n_1 n_2 n_3)$; The proximal map of TSP-$k$ norm by calling Algorithm 13 includes:

  - Step 1: Computing $\text{fft}$ takes $O(n_1 n_2 n_3 \log(n_3))$;
  
  - Step 2-4: Compute $n_3$ full SVD of $n_1 \times n_2$ matrices takes $O(n_1 n_2 n_3 \min\{n_1, n_2\})$;
  
  - Step 5: Sorting a vector of length $\min\{n_1, n_2\} n_3$ takes:
    
    $O(\min\{n_1, n_2\} n_3 \log(\min\{n_1, n_2\} n_3))$;
  
  - Step 6-7: Repeating binary search at most $k$ and $\min\{n_1, n_2\} n_3 - k$ times for ranges of $[1, k]$ and $[k, \min\{n_1, n_2\} n_3]$ each takes $O(k \log(k))$ and $(\min\{n_1, n_2\} n_3 - k)O(\min\{n_1, n_2\} n_3 - k)$, correspondingly;
  
  - Step 8: Element-wise vector arithmetic takes $O(\min\{n_1, n_2\} n_3)$;
  
  - Step 9: Rearranging vector elements back according to $\text{id}x$ kept during $\text{sort}$ operation takes $O(\min\{n_1, n_2\} n_3)$;
  
  - Step 10-12: $n_3$ matrix multiplications with the middle diagonal matrix of size $\min\{n_1, n_2\} \times \min\{n_1, n_2\}$ take $O(n_1 n_2 n_3 \min\{n_1, n_2\})$;
  
  - Step 13: $\text{ifft}$ takes $O(n_1 n_2 n_3 \log(n_3))$;

• Step 3: Element-wise addition/subtraction and soft-thresholding take $O(n_1 n_2 n_3)$;

• Step 4: Element-wise addition/subtraction take $O(n_1 n_2 n_3)$.
C.3 Proof of Materials in Section 6.5

C.3.1 Proof of Proposition 6.5.1

Proof. Beginning with the Lagrangian dual reformulation, the following sequence of equivalence relationship holds,

\[
\begin{align*}
\max_J \min_{\mathcal{E}} \frac{1}{2} \left\| \mathcal{L} \right\|_{\text{lisp}, k}^2 + \langle J, \mathcal{M}(\mathcal{L}) + \mathcal{E} - \mathcal{M}(\mathcal{X}) \rangle \\
\Leftrightarrow \max_J \left[ \min_{\mathcal{L}} \left( \frac{1}{2} \left\| \mathcal{L} \right\|_{\text{lisp}, k}^2 + \langle J, \mathcal{M}(\mathcal{L}) \rangle \right) \right] \\
\Leftrightarrow \max_J \left[ \min_{\mathcal{L}} - \left( \langle -\mathcal{M}^\top(J), \mathcal{L} \rangle - \frac{1}{2} \left\| \mathcal{L} \right\|_{\text{lisp}, k}^2 \right) \right. \\
\left. + \min_{\| \mathcal{E} \|_s \leq \tau} \left( \langle -J, \mathcal{E} \rangle - \langle J, \mathcal{M}(\mathcal{X}) \rangle \right) \right] \\
\Leftrightarrow \max_J \left[ - \max_{\mathcal{L}} \left( \langle -\mathcal{M}^\top(J), \mathcal{L} \rangle - \frac{1}{2} \left\| \mathcal{L} \right\|_{\text{lisp}, k}^2 \right) \right. \\
\left. - \max_{\| \mathcal{E} \|_s \leq \tau} \left( \langle -J, \mathcal{E} \rangle - \langle J, \mathcal{M}(\mathcal{X}) \rangle \right) \right] \\
\Leftrightarrow \max_J - \left[ \frac{1}{2} \left\| -\mathcal{M}^\top(J) \right\|_{\text{lisp}, k}^* + \langle J, \mathcal{M}(\mathcal{X}) \rangle + \tau \| J \|_s \right] \\
\Leftrightarrow - \min_J \left[ \frac{1}{2} \left\| -\mathcal{M}^\top(J) \right\|_{\text{lisp}, k}^* + \langle J, \mathcal{M}(\mathcal{X}) \rangle + \tau \| J \|_s \right].
\end{align*}
\]

(C.3.20)

In above, (i) is by the definition of Fenchel conjugate of 
\[ \frac{1}{2} \| \cdot \|^2_{\text{lisp}, k} \], i.e.

\[
\max_{\mathcal{L}} \left( \langle -\mathcal{M}^\top(J), \mathcal{L} \rangle - \frac{1}{2} \left\| \mathcal{L} \right\|_{\text{lisp}, k}^2 \right) = \frac{1}{2} \left\| -\mathcal{M}^\top(J) \right\|_{\text{lisp}, k}^*; \tag{C.3.21}
\]

and (ii) is by the definition the dual norm of \( \| \cdot \|_s \), i.e. (ii) = \( \tau \| J \|_s \). As a result, we can equivalently solve the dual objective,

\[
\min_J \mathcal{D}(J) := \min_J f(J) + h(J). \tag{C.3.22}
\]

\[\square\]
C.3.2 Proof of Proposition 6.5.2

Proof. By taking the (sub)gradient of $f(\Gamma)$ in eq.(12), we have

$$g(\mathcal{J}) = \partial \left( \frac{1}{2} (\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2 \right) + \mathcal{M}(\chi). \quad \text{(C.3.23)}$$

By the relation of eq.(C.3.21), it gives

$$\partial \left( \frac{1}{2} (\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2 \right)$$

$$= \partial \left( \max_{\mathcal{L}} (\langle -\mathcal{M}^T(\mathcal{J}), \mathcal{L} \rangle - \frac{1}{2} \| \mathcal{L} \|^2_{tsp,k}) \right)$$

$$= \partial (\langle -\mathcal{M}^T(\mathcal{J}), \mathcal{L}^\# \rangle - \frac{1}{2} \| \mathcal{L}^\# \|^2_{tsp,k})$$

$$= - \mathcal{M}(\mathcal{L}^\#), \quad \text{(C.3.24)}$$

where the last equality is because the (sub)gradient is taken with respect to $\mathcal{J}$. The optimum $\mathcal{L}^\#$ in the second equality should satisfy,

$$\mathcal{L}^\# = \arg \max_{\mathcal{L}} (\langle -\mathcal{M}^T(\mathcal{J}), \mathcal{L} \rangle - \frac{1}{2} \| \mathcal{L} \|^2_{tsp,k}), \quad \text{(C.3.25)}$$

which has closed-form solution for every norm function. We provide the details for our TSP-$k$ norm in the following for completeness. In order to compute $\mathcal{L}^\#$, we have

$$\langle -\mathcal{M}^T(\mathcal{J}), \mathcal{L} \rangle - \frac{1}{2} \| \mathcal{L} \|^2_{tsp,k}$$

$$\leq (\| \mathcal{L} \|^*_{tsp,k} \cdot \| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k} - \frac{1}{2} \| \mathcal{L} \|^2_{tsp,k})$$

$$= - \frac{1}{2} (\| \mathcal{L} \|^*_{tsp,k} \cdot \| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2 + \frac{1}{2} (\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2$$

$$\leq \frac{1}{2} (\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2, \quad \text{(C.3.26)}$$

where both inequality are obtained at $\mathcal{L}$ satisfying $\| \mathcal{L} \|^*_{tsp,k} = \| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k}$. Hence, the “scale” of $\mathcal{L}^\#$ under TSP-$k$ norm is $\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k}$. Fixing the scale, we need to decide the “direction” of $\mathcal{L}^\#$, i.e., a tensor $\mathcal{A}^\#$ with unit TSP-$k$ norm and the maximization becomes,

$$\arg \max_{\| \mathcal{A} \|^*_{tsp,k} \leq 1} \langle -\mathcal{M}^T(\mathcal{J}), \mathcal{A} \rangle - \mathcal{M}^T(\mathcal{J}) \| \mathcal{L}^\# \|^*_{tsp,k} + \frac{1}{2} (\| - \mathcal{M}^T(\mathcal{J}) \|^*_{tsp,k})^2, \quad \text{(C.3.27)}$$

which is the polar operator of TSP-$k$ norm at $\mathcal{J} = -\mathcal{M}^T(\mathcal{J})$:

$$\mathcal{A}^\# = \arg \max_{\| \mathcal{A} \|^*_{tsp,k} \leq 1} \langle -\mathcal{M}^T(\mathcal{J}), \mathcal{A} \rangle. \quad \text{(C.3.28)}$$
As a result, we have proved that
\[
\mathcal{L}^\# = \| - \mathcal{M}^\top(\mathcal{J}) \|_{\text{tsp}, k}^* \cdot \mathcal{A}^\#.
\]  
\hfill (C.3.29)

\section*{C.3.3 Proof of Proposition 6.5.3}

\textit{Proof.} We proceed the derivation with the FFT transformed block diagonal matrix, which gives the equivalent formulation of eq.(41) in the paper in Fourier domain as
\[
\mathcal{A}^\# = \arg\max_{n_3} \frac{1}{n_3} \| \mathcal{T}, \mathcal{A} \|_{\text{tsp}, k}^*.
\]  
\hfill (C.3.30)

The maximum is obtained when \( \mathcal{A}^\# / n_3 \) shares the same \( \mathcal{U}, \mathcal{V} \) of \( \mathcal{T} \), which further converts computation to
\[
\sigma_{\mathcal{A}^\# / n_3}^\perp = \arg\max_{\| v \|_{\text{sp}, k} \leq 1} (v, \sigma_{\mathcal{T}}^\perp).
\]  
\hfill (C.3.31)

The above eq.(C.3.31) is the polar operator of the vector \( k \)-support norms which ensures \( v \) the following closed-form computation
\[
(\sigma_{\mathcal{A}^\# / n_3}^\perp)_j = v_j = \begin{cases} 
\frac{(\sigma_{\mathcal{T}}^\perp)_j}{\| (\sigma_{\mathcal{T}}^\perp)_{[1:k]} \|_2}, & j \in [1 : k], \\
0, & j \in [k : D].
\end{cases}
\]  
\hfill (C.3.32)

Hence, after reshuffling the elements of \( \sigma_{\mathcal{A}^\#}^\perp \) back to their position in the original frontal slices according to \( \text{id} \) kept during the \texttt{sort} operation, the polar map is proved to take the form as in eq.(42) and (43) in the paper. \hfill \Box

\section*{C.3.4 Proof of Corollary 6.5.1}

\textit{Proof.} To prove the Corollary, it suffices to show the singular values of \( \mathcal{L}^\# \). With \( \mathcal{T} = -\mathcal{M}^\top(\mathcal{J}) \), note that \( \| \mathcal{T} \|_{\text{tsp}, k}^* = \| \sigma_\mathcal{T} \|_{\text{sp}, k}^* = \| (\sigma_\mathcal{T}^\perp)_{[1:k]} \|_2 \). By eq.(C.3.29) and (C.3.32) and the linearity of \texttt{fft}, the singular values \( \sigma_{\mathcal{L}^\#}^\perp \) are
\[
(\sigma_{\mathcal{L}^\#(\text{id} \text{x})})_j = \begin{cases} 
n_3(\sigma_{\mathcal{T}}^\perp)_j, & j \in [1 : k], \\
0, & j \in [k : D].
\end{cases}
\]  
\hfill (C.3.33)

The remaining is followed by Proposition V.3. \hfill \Box
C.3.5 Line-search Subroutine

Ideally, the algorithm performs better when adapting to the local smoothness of the dual loss function. To investigate such possibility, we study the structure of the (sub)gradient set. As the key ingredient of the (sub)gradient is the differentiation of the dual tensor spectral $k$-support norm, we describe its structure in the following proposition.

In particular, let $y = -M^T(J)$. We describe the (sub)gradient set of $\|y\|_{tsp,k}^*$ in the following.

**Proposition C.3.1.** Denote a particular tensor SVD of $Y$ by $Y = U \cdot S V^T$ and the associated Fourier transformed diagonal matrix by $\hat{Y} = \hat{U}\text{diag} (\sigma) \hat{V}^T$. Denote the sorted non-increasing singular vectors by $\sigma_i^i$. Assume the sorted singular values satisfy

$$\sigma_1^i \geq ... > \sigma_{k-a+1}^i = ... = \sigma_{k-b}^i > ... \geq \sigma_{n_2 \cdot n_3}^i.$$  \hspace{1cm} (C.3.34)

Let $\mathbb{K}_1$ denote the corresponding index set in $\sigma$ of entries in $\sigma_{[1:k-a]}^i$, and $\mathbb{K}_2$ for the indices of entries in $\sigma_{[k-a+1:k+b]}^i$, and $\mathbb{K}$ denotes the indices of the leading $k$ singular values. The (sub)gradient set of the dual spectral $k$-support norm at $\hat{Y}$ is denoted by $\hat{G}$, of which each $i$-th frontal slice in the Fourier domain $\hat{G}^{(i)} = \frac{1}{\|\hat{Y}\|_{tsp,k}^*} \left\{ \sum_{k_1 \in \mathbb{K}_1} \hat{U}_{k_1}^i \sigma_{k_1}^i (\hat{V}_{k_1}^{(i)})^\top + \sum_{k_2 \in \mathbb{K}_2} \hat{U}_{k_2}^i T (\hat{V}_{k_2}^{(i)})^\top \right\},$ \hspace{1cm} (C.3.35)

where $T$ satisfies $T = T^\top$, $\|T\|_2 \leq \sigma_k^i$ and $\|T\|_* = a \sigma_k^i$. In particular, $\|y\|_{tsp,k}^*$ is differentiable at $y$, if $\sigma_k^i > \sigma_{k+1}^i$ or $\sigma_k^i = 0$, during when $\hat{G}$ denotes the unique gradient, which satisfies,

$$\hat{G}^{(i)} = \frac{1}{\|\hat{Y}\|_{tsp,k}^*} \sum_{k \in \mathbb{K}} \hat{U}_k^i \sigma_k^i (\hat{V}_k^{(i)})^\top.$$  \hspace{1cm} (C.3.36)

The singular value subsequence of $(i)$ in eq.(C.3.34) is the longest subsequence which equals to each other containing the $k$-the largest singular value. Under the condition $\sigma_k^i > \sigma_{k+1}^i$ or $\sigma_k^i = 0$, the leading $k$ singular values are well-separated.
with the remaining smaller singular values starting from \( k + 1 \)-th entry. Also, eq. (C.3.36) is always a particular choice in eq. (C.3.35), since eq. (C.3.36) corresponds to \( T = \sigma_k^j \text{diag}(1_{[1:a]}) \) (obviously \( \|T\|_2 \leq \sigma_k^j \) and \( \|T\|_* = a\sigma_k^j \) are satisfied). In particular, when the singular value are “well-separated”, eq. (C.3.36) becomes the unique element of the subgradient set of eq. (C.3.35), i.e. the dual tensor spectral \( k \)-support norm is differentiable at \( \mathbf{y} \) with “well-separated” singular values under t-SVD. Finally, by the computation of the polar operator and the dual (sub)gradient, we actually always choose eq. (C.3.36) in the dual objective optimization.

Despite the smoothness variation, we always choose eq. (C.3.36) during the dual (sub)gradient computation. Hence, it would be desirable the our optimization procedure would adaptive to the smoothness to the dual loss function. The \( \nu \geq 1 \) is called the Hölder smoothness order, and the associated parameter \( H_\nu \) is defined as

\[
H_\nu = H_\nu(f) := \sup_{x_1 \neq x_2} \left\{ \frac{||\nabla f(x_1) - \nabla f(x_2)||_2}{||x_1 - x_2||_2^\nu} \right\}, \quad (C.3.37)
\]

where \( \nabla f(x) \) denotes a (sub)gradient of \( f \) at \( x \).

In particular, we propose to rely on the concept of the Hölder smoothness \( \nu \in [0,1] \) and the line-search strategy that adapts to this smoothness variation. By Hölder smoothness, the smooth case corresponds to \( \nu = 1 \) in the Hölder smoothness, while when the (sub)gradient set is not unique, it corresponds to \( \nu = 0 \) in the Hölder smoothness. \( H_t \) is determined a backtracking line search, which searches the \( J_{t+1} \) satisfying,

\[
f(J_{t+1}) + h(J_{t+1}) \\
\leq f(\tilde{J}_t) + \langle a(\tilde{J}_t), J_{t+1} - \tilde{J}_t \rangle + \frac{H_t}{2} ||J_{t+1} - \tilde{J}_t||_F^2 + h(J_{t+1}) \quad (C.3.38)
\]

Algorithm 22 presents the backtracking line search for determining \( H_t \) and updating \( J_{t+1} \). The dominating per-iteration complexity, with the remaining being simple tensor inner product and element-wise operation, lines in Line 3 and 4: 1) proximal mapping with respect to the dual regularizer and 2) the computation of \( J_{t,i+1} \). The former computational cost mainly comes from the projection onto the unit \( || \cdot ||_1 \) ball which has \( O(n_1n_2n_3) \) complexity algorithm. The latter mainly
needs to compute the dual tensor spectral $k$-support norm on $\mathbf{J}_{t,i+1}$, which relies on the leading $k$ singular values of each $\tilde{\mathbf{J}}_{t,i+1}$ and can be obtained in $O(kn_1n_2n_3)$ computation by partial SVD. Note that such line search is only possible after our dual reformulation, because the primal norm computation during such backtracking would require full SVD which cost super-linear complexity $O(n_1(n_2)^2n_3)$. Also, by [149], the line search requires roughly two rounds on average. In sum, the line search step takes the complexity $O(kn_1n_2n_3)$.

### C.3.6 Computational Complexity Analysis for Algorithm 16

In each iteration, the computational complexity is as follows:

- **Line 2 $O((k + \log n_3)n_1n_2n_3)$**: This step computes the atom by polar operator for (sub)gradient evaluation, which involves one fft and one ifft operation cost $O(n_1n_2n_3 \log n_3$, plus $n_3$ $k$-SVD costing $O(kn_1n_2n_3)$;

- **Line 3 $O(n_1n_2n_3)$**: This step computes the (sub)gradient of the dual loss based on the atom by eq.(32), which are element-wise mapping and addition of tensor
entries taking $O(n_1n_2n_3)$ computation;

- Line 4 $O(n_1n_2n_3)$: This step searches for the smoothness parameter $H_t$. Note that to evaluate $Q_{H_t}$, as terms like SVD decomposition are already computed in previous step, it only need to re-compute inner product and addition, which takes $O(n_1n_2n_3)$ at most;

- Line 8 and 9 $O(n_1n_2n_3)$: These two steps update the dual interpolation variable and primal variable correspondingly, which are element-wise addition costing $O(n_1n_2n_3)$. 
Appendix D

Proofs of Materials in Chapter 7

This chapter provides proofs for the main results of the paper. Specifically, Section D.1 recalls additional definition and techniques. Section D.2 contains proofs for BCFW under arbitrary sampling, and Section D.3 describes the detailed differentially private BCFW for feature-wise distributed dataset and proves the privacy and utility results for this algorithm.

D.1 Preliminary

For completeness, this section collects additional definitions and technical tools.

Definition D.1.1. (Normalized leading eigenvalue) For a \( d \times d \) matrix \( M \), the normalized leading eigenvalue \( \sigma'(M) \) is

\[
\sigma'(M) = \max_{h \in \mathbb{R}^d} \{ h^T M h, \text{ s.t. } h^T \text{diag}(M) h \leq 1 \}, \tag{D.1.1}
\]

where \( \text{diag}(M) \) is the diagonal matrix of \( M \).

Lemma D.1.2. (Expectation over a random index set) Let \( S \) be a proper sampling distribution with probability for each index \( p = \{ p_1, p_2, ..., p_d \} \) and denote the pairwise probability matrix by \( P \). Assume the index set \( T \) is sampled according to \( S \), for any vector \( x, h \in \mathbb{R}^d \), matrix \( M \in \mathbb{R}^{d \times d} \), the following expectations over \( T \) hold,
1. \(\mathbb{E}[\langle x, h_{[T]} \rangle] = \sum_{i=1}^{d} p_i x_i h_i;\)

2. \(\mathbb{E}[h_{[T]}^T M h_{[T]}] = h^T (P \circ M) h.\)

Lemma D.1.3. (Serial composition for differential privacy \[33\]) Let \(\mathcal{ALG}_1\) be \((\epsilon_1, \delta_1)\)-differential privacy and \(\mathcal{ALG}_2\) be \((\epsilon_2, \delta_2)\)-differential privacy. Then, their combination is \((\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)\)-differential privacy.

Lemma D.1.4. (Advanced composition for differential privacy \[33\]) For all \(\epsilon, \delta_1, \delta_2 \geq 0\), the class of \((\epsilon, \delta_1)\)-differentially private mechanisms satisfies \((\epsilon', k\delta_1 + \delta_2)\)-differential privacy under \(k\)-fold adaptive composition for:

\[
\epsilon = \sqrt{2k \ln(1/\delta_2)} \epsilon + k\epsilon(\epsilon' - 1). \tag{D.1.2}
\]

Since \(k\epsilon(\epsilon' - 1) \leq k\epsilon^2 = O(\epsilon)\), which is small compared to \(\sqrt{2k \ln(1/\delta_2)} \epsilon\), the advanced composition roughly provides \((\sqrt{2k \ln(1/\delta_2)} \epsilon, k\delta_1 + \delta_2)\)-differential privacy.

Definition D.1.5. (Report-Noisy-Max) Given a dataset \(\mathbb{D}\), denote a collection of \(P\) functions \(v_1, \ldots, v_p, \ldots, v_P\) defined on \(\mathbb{D}\) with \(\ell_1\)-sensitivity \(\xi\). The Report-Noisy-Max selects the index whose perturbed value function is the maximum. That is: \(\hat{p}^* = \arg\max_{p \in [P]} \hat{v}_p\), where \(\hat{v}_p = v_p + \text{pert}\) with \(\text{pert} \sim \text{Lap}(\xi)\) for \(p \in [P]\).

Lemma D.1.6. (Privacy guarantee for Report-Noisy-Max mechanism \[33\]) The Report-Noisy-Max mechanism preserves \((2\epsilon, 0)\)-differential privacy.

Lemma D.1.7. (JL-transform lemma \[137\]) A random matrix \(J \in \mathbb{R}^{m \times n}\) forms a JL-transform with parameter \(\iota\), for all \(a, a'\), with probability at least \(1 - \delta\), it holds that \(|\langle Ja, Ja' \rangle - \langle a, a' \rangle| \leq \iota \|a\|_2 \|a'\|_2\).

Lemma D.1.8. (Privacy guarantee of private JL-transform \[70\]) Let \(J : m \times n\) be a Gaussian sketching matrix and the entries of noise matrix \(N : m \times d\) be drawn by \(\text{pert} \sim \mathcal{N}(0, \pi^2)\), with \(\pi \geq \sigma(J) \sqrt{2 \ln(1/2\delta) + \epsilon}/\epsilon\). Then for matrix \(A : n \times d\) with each row \(a\) satisfies \(\|a\|_2 \leq 1\), \(JA + N\) is \((\epsilon, \delta)\)-differentially private.
D.2 Proofs of Materials in Section 7.3

D.2.1 Proof of Proposition 7.3.2

Proof. Recall the assumption,
\[ f(y) \leq f(x) + (y - x, \nabla f(x)) + \frac{1}{2} (y - x)^\top A^\top A(y - x). \quad (D.2.3) \]
By substituting \( y = x + \gamma(s^{[\tau]} - x^{[\tau]}) \) in and rearranging terms, we have
\[ \frac{2}{\gamma^2} (f(x + \gamma(s^{[\tau]} - x^{[\tau]})) - f(x) - \gamma(s^{[\tau]} - x^{[\tau]}, \nabla_{(\tau)} f(x))) \leq (s^{[\tau]} - x^{[\tau]})^\top A^\top A(s^{[\tau]} - x^{[\tau]}). \quad (D.2.4) \]
Taking expectations according to the sampling distribution \( S \) on both sides conditioned on proper \( x, s, \gamma \), then taking supremum on both sides with respect to \( x, s, \gamma \), we have
\[ C_{f,S}^E \leq \sup_{x, s, \gamma} \mathbb{E} [(s^{[\tau]} - x^{[\tau]})^\top A^\top A(s^{[\tau]} - x^{[\tau])}], \quad (D.2.5) \]
where the left-hand-side is by the definition of the expected curvature. Recall the \( d \times d \) probability matrix \( P \) of the sampling distribution \( S \) whose \((i, j)\)-th element \( P_{ij} \) is the probability of the \( i \)-th and \( j \)-th block sampled simultaneously. Then, by Lemma\[D.1.2\] we have
\[ \mathbb{E} [(s^{[\tau]} - x^{[\tau]})^\top A^\top A(s^{[\tau]} - x^{[\tau])}] = (s - x)^\top (P \circ (A^\top A))(s - x). \quad (D.2.6) \]
By choosing \( \beta = (\beta_1, \ldots, \beta_d) \), where \( \beta_i = \min\{\sigma'(P), \sigma'(A^\top A)\} A_i \|A_i\|_2^2 \), where \( \sigma'(P), \sigma'(A^\top A) \) are the largest normalized eigenvalue of the matrices \( P \) and \( A^\top A \), we have
\[ (s - x)^\top (P \circ (A^\top A))(s - x) \leq \sum_{i=1}^d \beta_i p_i \|s_i - x_i\|_2^2 \quad (D.2.7) \]
Taking supremum on both sides, we have
\[ C_{f,S}^E \leq \sup_{x, s \in M} \sum_{i=1}^d \beta_i p_i \|s_i - x_i\|_2^2 = \sum_{i=1}^d p_i \beta_i D_{M_i}^2. \quad (D.2.8) \]
Hence, the expected curvature is less than or equal to the “expected” coordinate “Lipschitz smoothness” parameter times the coordinate squared diameter of the domain. The right hand side gives a worst case approximation for the expected curvature under arbitrary sampling. \(\square\)
D.2.2 Proof of Proposition 7.3.3

Proof. 1. \(C_f^{\text{element}} = C_f\) is by definition;

2. To prove the relationship \(C_f^{\text{uni-seri}} \leq \frac{1}{d} C_f^{\circ}\):

\[
C_f^{\circ} = \sum_{i=1}^{d} C_f^{i} = d \sum_{i=1}^{d} \frac{1}{d} C_f^{i}
\]

\[
= d \mathbb{E} \left[ \sup_{x, s \in \mathcal{M}, \gamma \in [0,1]} \frac{\gamma}{2} (f(x + \gamma(s_i - x_i)) - f(x) - \gamma\langle s_i - x_i, \nabla_i f(x) \rangle) \right]
\]

\[
\geq d \mathbb{E} \left[ \frac{2}{\gamma^2} (f(x + \gamma(s_i - x_i)) - f(x) - \gamma\langle s_i - x_i, \nabla_i f(x) \rangle) \right].
\]

Taking supremum on both sides, the left-hand side is constant \(C_f^{\circ}\) and the right-hand-side is \(dC_f^{\text{uni-seri}}\), which proves the relationship.

3. To prove the relationship \(C_f^{\text{nice}} \leq C_f^{\circ}\):

\[
C_f^{\circ} = \left( \frac{d}{\tau} \right)^{-1} \sum_{S \subseteq [d], |S| = \tau} C_f^{(S)}
\]

\[
= \sum_{S \subseteq [d], |S| = \tau} \left( \frac{d}{\tau} \right)^{-1} \mathbb{E} \left[ \sup_{x, s \in \mathcal{M}, \gamma \in [0,1], |S| = \tau} \frac{2}{\gamma^2} (f(x + \gamma(s|S| - x|S|)) - f(x) - \gamma\langle s|S| - x|S|, \nabla|S| f(x) \rangle) \right]
\]

\[
\geq \sum_{S \subseteq [d], |S| = \tau} \left( \frac{d}{\tau} \right)^{-1} \frac{2}{\gamma^2} (f(x + \gamma(s|S| - x|S|)) - f(x) - \gamma\langle s|S| - x|S|, \nabla|S| f(x) \rangle)
\]

\[
= \mathbb{E} \left[ \frac{2}{\gamma^2} (f(x + \gamma(s|S| - x|S|)) - f(x) - \gamma\langle s|S| - x|S|, \nabla|S| f(x) \rangle) \right].
\]

(D.2.9)

Taking supremum on both sides, the left-hand-side is constant \(C_f^{\circ}\) and the right-hand-side is \(C_f^{\text{nice}}\), which proves the relationship.

\[\square\]

D.2.3 Proof of Proposition 7.3.4

Proof. By eq. (D.2.5) and eq. (D.2.6), we have

\[
C_f^{\text{nice}} \leq \sup_{x, s \in \mathcal{M}} [(s - x)^\top (\mathbb{P}_{\text{nice}} \circ (A^\top A))(s - x)],
\]

(D.2.11)

where the probability matrix is

\[
\mathbb{P}_{ij} = \begin{cases} \frac{\tau}{d} & i = j, \\ \frac{\tau(\tau-1)}{d(d-1)} & i \neq j. \end{cases}
\]

(D.2.12)
\[
\sup_{\mathbf{s}, \mathbf{x} \in \mathcal{M}} [(\mathbf{s} - \mathbf{x})^\top (\mathbb{P}_{\text{nice}} \circ (\mathbf{A}^\top \mathbf{A}))(\mathbf{s} - \mathbf{x})]
\]
\[
= \sup_{\mathbf{s}, \mathbf{x} \in \mathcal{M}} \left[ \sum_{i=1}^d \frac{\tau}{d} ||\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_i)||_2^2 + \sum_{i,j \in [d], i \neq j} \frac{\tau(\tau - 1)}{d(d - 1)} (\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_j))^\top (\mathbf{A}_j(\mathbf{s}_i - \mathbf{x}_j)) \right]
\]
\[
\leq \tau \mu_1 + \tau(\tau - 1)\mu_2,
\]
(D.2.13)

where the inequality is by the sub-additive property of the supremum operation and the definition of \(\mu_1\) and \(\mu_2\).

\[\square\]

D.2.4 Proof of Proposition 7.3.5

Proof. By eq. (D.2.5) and eq. (D.2.6), we have

\[
C_{f}^{E(K, \tau)} \leq \sup_{\mathbf{x}, \mathbf{s} \in \mathcal{M}} [(\mathbf{s} - \mathbf{x})^\top (\mathbb{P}_{(K, \tau)} \circ (\mathbf{A}^\top \mathbf{A}))(\mathbf{s} - \mathbf{x})],
\]
(D.2.14)

where \(\mathbb{P}_{(K, \tau)}\) takes the following value according to (note that we consider \(d/K > 1\) case (i.e. each node contains ore than one feature)):

\[
\mathbb{P}_{ij} = \begin{cases} 
\frac{\tau}{d/K} := \theta_1 & i = j, \\
\frac{\tau(\tau - 1)}{d(K(d/K - 1))} := \theta_2 & i \neq j, i, j \text{ in the same node}, \\
\frac{\tau^2}{(d/K)^2} := \theta_3 & i \neq j, i, j \text{ in the different nodes}.
\end{cases}
\]

(D.2.15)

Then, we have the following estimation for the upper bound of \(C_{f}^{E(K, \tau)}\):

\[
C_{f}^{E(K, \tau)} \leq \sup_{\mathbf{x}, \mathbf{s} \in \mathcal{M}} \left[ \theta_1 \sum_{i=1}^d ||\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_i)||_2^2 + \theta_2 \sum_{k=1}^K \sum_{i,j \in \mathcal{P}_k, i \neq j} (\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_j))^\top (\mathbf{A}_j(\mathbf{s}_i - \mathbf{x}_j)) \right]
\]
\[
+ \theta_3 \sum_{i \in \mathcal{P}_{k_1}, j \in \mathcal{P}_{k_2}, k_1, k_2 \in [K], k_1 \neq k_2} (\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_j))^\top (\mathbf{A}_j(\mathbf{s}_i - \mathbf{x}_j)) \right]
\]
\[
\leq K\tau \mu_1 + K\tau(\tau - 1)\mu_2 + K(K - 1)\tau^2 \mu_3,
\]
(D.2.16)

where \(\mu_1 = \sup_{i \in [d]} ||\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_i)||_2^2\), \(\mu_2 = \sup_{i,j \in [d], i \neq j} (\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_j))^\top (\mathbf{A}_j(\mathbf{s}_i - \mathbf{x}_j))\), \(\mu_3 = \sup_{i \in \mathcal{P}_{k_1}, j \in \mathcal{P}_{k_2}, k_1 \neq k_2} (\mathbf{A}_i(\mathbf{s}_i - \mathbf{x}_j))^\top (\mathbf{A}_j(\mathbf{s}_i - \mathbf{x}_j))\).
D.2.5 Proof of Theorem 7.3.6

Proof. We begin with the definition of \( x^{t+1} \) in the step 6 of algorithm 1. In the following, (eq. D.2.17 (i)) is by the definition of the expected curvature; (eq. D.2.17 (ii)) is by bounding the term \( (I) \) in eq. (D.2.17), which is based on the assumption on the inexact gradient \( \hat{\nabla} f(x^t) \) and the inexact oracle \( \hat{s}^t \); (eq. D.2.17 (iii)) is by the definition of the exact linear oracle \( s^t \). With all expectations taken with respect to the sampling \( T^t \) and conditioned on \( \{x^t, ..., x^0\} \), we have

\[
\mathbb{E}[f(x^{t+1})] = \mathbb{E}[f(x^t + \gamma^t(\hat{s}^t_{[T^t]} - x^t_{[S^t]}))] \\
\leq (i) \quad \mathbb{E}[f(x^t + \gamma^t(\hat{s}^t_{[T^t]} - x^t_{[T^t]}), \nabla f(x^t))] + \frac{(\gamma^t)^2}{2} C^E f \\
= f(x^t) + \gamma^t \sum_{i \in T^t} \mathbb{E}[(\hat{s}^t_{[i]} - s^t_{[i]}, \nabla f(x^t))] + \gamma^t \sum_{i \in T^t} \mathbb{E}[(\hat{s}^t_{[i]} - x^t_{[i]}, \nabla f(x^t))] + \frac{(\gamma^t)^2}{2} C^E f \\
\leq (ii) \quad f(x^t) + \gamma^t \mathbb{E}[(\hat{s}^t_{[T^t]} - x^t_{[T^t]}, \nabla f(x^t))] + \frac{(\gamma^t)^2}{2} C^E f \\
\leq (iii) \quad f(x^t) + \gamma^t \mathbb{E}[(x^t_{[T^t]} - x^t_{[T^t]}, \nabla f(x^t))] + \frac{(\gamma^t)^2}{2} (1 + \varrho) C^E f, \\
\tag{D.2.17}
\]

where the upper bound of term \( (I) \) is detailed as follows. Recall the definitions of the terms:

- \( s^t \) is the exact linear oracle computed with exact gradient \( \nabla f(x^t) \), where each block within \( T^t \) satisfies:

\[
\hat{s}^t_{(i)} = \arg \min_{s^t(\cdot) \in \mathcal{M}_i} \langle s^t_{(i)}, \nabla_{(i)} f(x^t) \rangle, \\
\tag{D.2.18}
\]

which implies, by the definition of the dual norm of \( \|\cdot\|_{\mathcal{M}_i} \), that \( \langle s^t_{(i)}, \nabla_{(i)} f(x^t) \rangle = \|\nabla_{(i)} f(x^t)\|_{\mathcal{M}_i}^* \) where \( \| \cdot \|_{\mathcal{M}_i}^* \) denotes the dual norm;

- \( \hat{s}^t \) is the exact linear oracle computed with the inexact gradient \( \hat{\nabla} f(x^t) \). Similar to the \( s^t \) case,

\[
\hat{s}^t_{(i)} = \arg \min_{s^t(\cdot) \in \mathcal{M}_i} \langle s^t_{(i)}, \hat{\nabla}_{(i)} f(x^t) \rangle, \\
\tag{D.2.19}
\]

217
which gives \( \langle \hat{s}_{(i)}^t, \hat{\nabla}_{(i)} f(x^t) \rangle = \| \hat{\nabla}_{(i)} f(x^t) \|_{\mathcal{M}_i}^* \); By the inexact gradient assumption,
\[
\| \nabla_{(i)} f(x^t) - \hat{\nabla}_{(i)} f(x^t) \|_{\mathcal{M}_i}^* \leq \frac{\vartheta_\alpha }{2} \gamma^t C_S^{\mathcal{F}_i}; \tag{D.2.20}
\]

- \( \hat{s}^t \) is the inexact linear oracle computed with the inexact gradient \( \hat{\nabla} f(x^t) \),
which gives
\[
\langle \hat{s}_{(i)}^t, \hat{\nabla}_{(i)} f(x^t) \rangle \leq \langle \hat{s}_{(i)}^t, \hat{\nabla}_{(i)} f(x^t) \rangle + \frac{\vartheta_\alpha }{2} \gamma^t C_S^{\mathcal{F}_i}. \tag{D.2.21}
\]

Then, term (I) is bounded as
\[
\sum_{i \in \mathcal{T}^t} \mathbb{E}[\langle \hat{s}_{[i]}^t - s_{[i]}^t, \nabla f(x^t) \rangle] = \mathbb{E}[\sum_{i \in \mathcal{T}^t} \langle \hat{s}_{[i]}^t, \nabla f(x^t) - \hat{\nabla} f(x^t) + \hat{\nabla} f(x^t) - \nabla f(x^t) \rangle - \langle s_{[i]}^t, \nabla f(x^t) \rangle]
= \mathbb{E}[\sum_{i \in \mathcal{T}^t} \langle \hat{s}_{[i]}^t, \nabla f(x^t) - \hat{\nabla} f(x^t) \rangle + \langle \hat{s}_{[i]}^t, \hat{\nabla} f(x^t) \rangle - \langle s_{[i]}^t, \nabla f(x^t) \rangle]
= \mathbb{E}[\sum_{i \in \mathcal{T}^t} \{ \langle \hat{s}_{[i]}^t, \nabla f(x^t) - \hat{\nabla} f(x^t) \rangle \} + \{ \langle \hat{s}_{[i]}^t - s_{[i]}^t, \hat{\nabla} f(x^t) \rangle \} + \{ \langle \hat{s}_{[i]}^t, \hat{\nabla} f(x^t) \rangle - \langle s_{[i]}^t, \nabla f(x^t) \rangle \}], \tag{D.2.22}
\]

where the terms in the curve blanket in the last inequality can be bounded as,

- The first blanket:
\[
\{ \langle \hat{s}_{[i]}^t, \nabla f(x^t) - \hat{\nabla} f(x^t) \rangle \} \leq \| \hat{s}_{[i]}^t \|_{\mathcal{M}_i} \| \nabla f(x^t) - \hat{\nabla} f(x^t) \|_{\mathcal{M}_i}^* \leq D_{\mathcal{M}_i} \frac{\vartheta_\alpha }{2} \gamma^t C_S^{\mathcal{F}_i} \tag{D.2.23}
\]

- The second blanket:
\[
\{ \langle \hat{s}_{[i]}^t - s_{[i]}^t, \hat{\nabla} f(x^t) \rangle \} \leq \frac{\vartheta_\alpha }{2} \gamma^t C_S^{\mathcal{F}_i} \tag{D.2.24}
\]

- The third blanket:
\[
\{ \langle \hat{s}_{[i]}^t, \nabla f(x^t) \rangle - \langle s_{[i]}^t, \nabla f(x^t) \rangle \} = \| \hat{\nabla} f(x^t) \|_{\mathcal{M}_i}^* - \| \nabla f(x^t) \|_{\mathcal{M}_i}^*
\leq \| \hat{\nabla} f(x^t) - \nabla f(x^t) \|_{\mathcal{M}_i}^* \leq \frac{\vartheta_\alpha }{2} \gamma^t C_S^{\mathcal{F}_i} \tag{D.2.25}
\]
Hence, collectively we have

$$\begin{align*}
(1) \leq \frac{(D_M + |T^t|)\varrho_g \gamma^t C_f^{ES}}{2} + \frac{\varrho_t |T^t| \gamma^t C_f^{ES}}{2} = \frac{\gamma^t C_f^{ES}}{2} \cdot ((D_M + |T^t|)\varrho_g + \varrho_t |T^t|) := \frac{\gamma^t C_f^{ES}}{2} \cdot \varrho,
\end{align*}$$

(D.2.26)

where $\varrho_g = \max_i \varrho_g^i$, $\varrho_t = \max_i \varrho_t^i$, and $|T^t|$ is the sampling size.

Next, computing the expectation for the term $(II)$ in eq.(D.2.17) (recall the vector $p = \{p_1, \ldots, p_d\}$ for representing the probability of sampling block $i$) by Lemma D.1.2 in eq.(D.2.27 (i)), and by the convexity of $f(x)$ eq.(D.2.27 (ii)), we have

$$\begin{align*}
\mathbb{E}[f(x^{t+1})] &\overset{(i)}{=} f(x^t) + \gamma^t \sum_{i=1}^{d} p_i (x^t_i - x^*_i, \nabla f(x^t)) + \frac{(\gamma^t)^2}{2} (1 + \varrho) C_f^{ES} \\
&\overset{(ii)}{\leq} f(x^t) + \gamma^t \sum_{i=1}^{d} p_i (f_i(x^*) - f_i(x^t)) + \frac{(\gamma^t)^2}{2} (1 + \varrho) C_f^{ES},
\end{align*}$$

(D.2.27)

where the equality eq.(D.2.27 (i)) for computing the expectation of the inner product over sampling set is justified by eq.(30) of Theorem 4 in [113].

Subtracting $f(x^*)$ from both sides and denoting the primal gap $f(x^t) - f(x^*)$ by $h(x^t)$, we have

$$\begin{align*}
\mathbb{E}[h(x^{t+1})] \leq h(x^t) - \gamma^t \sum_{i=1}^{d} p_i h_i(x^t) + \frac{(\gamma^t)^2}{2} (1 + \varrho) C_f^{ES} \\
&\leq (1 - \gamma^t p_{\min}) h(x^t) + \frac{(\gamma^t)^2}{2} (1 + \varrho) C_f^{ES} & \text{(D.2.28)}
\end{align*}$$

where $p_{\min}$ denotes the smallest sampling probability among the $d$ blocks and the last equality is by noting that the expectation is conditioned on $\{x^t, \ldots, x^0\}$.

Denote $C = p_{\min} \mathbb{E}[h(x^0)] + (1 + \varrho) C_f^{ES}$, let $\gamma^t = \frac{2}{p_{\min} t + 2}$, we show $\mathbb{E}[h(x^t)] \leq \frac{2 C}{p_{\min} t + 2}$ by induction. The initial condition holds with $\mathbb{E}[h(x^0)] \leq \frac{2 C}{p_{\min}^2}$, Assuming the $t$-th condition $\mathbb{E}[h(x^t)] \leq \frac{2 C}{p_{\min} t + 2}$ holds, by eq.(D.2.28) and substituting $\gamma^t$ in,
we have

$$\mathbb{E}[h(x^{t+1})] \leq \left(1 - \frac{2p_{min}}{p_{min}t + 2}\right) \frac{2C}{p_{min}t + 2} + \frac{4C}{2(p_{min}t + 2)^2}$$

$$= \frac{2C}{p_{min}t + 2} \left[1 - \frac{2p_{min}}{p_{min}t + 2} + \frac{p_{min}}{p_{min}t + 2}\right]$$

$$= \frac{2C}{p_{min}t + 2} \cdot \frac{p_{min}t + 2 - 2p_{min} + p_{min}}{p_{min}t + 2}$$

$$\leq \frac{2C}{p_{min}t + 2} \cdot \frac{p_{min}t + 2}{p_{min}t + p_{min} + 2}$$

$$= \frac{2C}{p_{min}(t + 1) + 2}.$$  \hspace{1cm} (D.2.29)

Hence, by induction, $$\mathbb{E}[f(x^t)] - f^* \leq \frac{2(p_{min}(f(x^0) - f^*) + (1+\delta)C_E^S)}{p_{min}t + 2p_{min}} = \frac{2(\mu(x^0) + (1+\delta)p_{min}C_E^S)}{p_{min}t + 2}. \hspace{1cm} \square$$

D.3 Proofs of Materials in Section 7.4

In this section, we consider $$(K, \tau)$$-distributed sampling with the features evenly randomly split on $$K$$ nodes. The constraint set is $$\eta$$-$l_1$$-norm ball, whose dual norm is $$l_{\infty}$$-norm. Thus, $$D_{M_k} = ||M_k||_1$$. Let $$G_k = ||(\nabla f_k) ||_\infty$$.

D.3.1 Lemma D.3.1 and Proof: Private feature sharing

**Lemma D.3.1.** Private active feature sharing preserves $$(\epsilon/2, \delta/2)$$-differential privacy.

**Proof.** Since at most $$KT$$ non-duplicate features are communicated over $$T$$ iterations, the matrix consisting of the transmitted features, denoted by $$A_{\text{trans}}$$, is no larger than $$n$$ by $$KT$$. With JL-transform matrix $$J : m \times n$$, and noise matrix $$N = [\xi_1^1, \ldots, \xi_K^1, \ldots, \xi_1^T, \ldots, \xi_K^T]$$ with each element of $$\xi_k^i : m \times 1$$ sampled from $$pert' \sim \mathcal{N}(0, \pi^2)$$, the privately transmitted features can be collectively represented as $$J(\frac{1}{n}A_{\text{trans}}) + N$$. Assuming a random data point $$D_i$$ is changed to $$D'_i$$, the $$\ell_2$$-sensitivity of $$\frac{1}{n}A_{\text{trans}}$$ can be bounded as

$$\frac{1}{n}||A_{\text{trans}} - A'_{\text{trans}}||_2 = \frac{1}{n}||D_{\text{i,trans}} - D'_{\text{i,trans}}||_2 \leq \frac{\sqrt{KT}}{n}.$$  \hspace{1cm} (D.3.30)
By the private JL-transform lemma, with the noise parameter \( \pi = \frac{\sigma(J)\sqrt{KT}\sqrt{2(\ln(\frac{1}{\varepsilon})+\varepsilon/2)}}{\sigma n/2} \), the privately transmitted features \( (J(\frac{1}{n}A_{\text{trans}}) + N) \) is \((\varepsilon/2, \delta/2))\)-DP.

\[ \square \]

**D.3.2 Lemma D.3.2 and Proof: Private index computing**

**Lemma D.3.2.** Private index computing preserves \((\varepsilon/2, \delta/2))\)-differential privacy

**Proof.** For user node \( k \), the perturbation \( \text{pert}_k \) is from Laplace distribution denoted as \( \text{pert}_k \sim \text{Lap}(\frac{G_k||M_k||_1/n\cdot2^{2T\log[1/(\delta/2)]}}{(\frac{1}{T})/2}) \). Hence, by the privacy guarantee of the Report-Noisy-Max, each node preserves \((\frac{\varepsilon_k}{\sqrt{2T\log(1/T)}}, 0))\)-DP for one iteration. By serial composition theorem for \( K \) nodes and by \( \sum_k \varepsilon_k = \varepsilon \) (\( \varepsilon_k \) is assigned based on local sensitivity, i.e. \( \varepsilon_k = \varepsilon \cdot \frac{G_k||M_k||_1}{\sum_{k=1}^{K}G_k||M_k||_1} \)), the algorithm preserves \((\frac{\varepsilon/2}{\sqrt{2T\log(1/T)}}, 0))\)-DP for one iteration. Then, by the strong composition theorem over \( T \) iterations, private index communicating is \((\varepsilon/2, \delta/2))\)-DP.

\[ \square \]

**D.3.3 Proof of Theorem 7.4.1**

**Proof.** Apply simple composition property of DP with Lemma 1 and Lemma 2, the algorithm is \((\varepsilon, \delta))\)-differentially private.

\[ \square \]

**D.3.4 Proof of Theorem 7.4.2**

**Proof.** The utility guarantee is based on the general BCFW convergence result in the previous section, tailored to private LASSO task with \((K, \tau))\)-distributed sampling. We first derive the inexact gradient parameter \( g_g \) and the inexact linear oracle parameter \( g_l \), based on the private feature sharing and private index communicating properties correspondingly. The aim is to represent \( g_g \) and \( g_l \) with the noise parameters of \( \text{pert} \) and \( \text{pert}' \) correspondingly.

- **Inexact partial gradient parameter \( g_g \):** Since \( \max_i ||\hat{\nabla}_i f(x^t) - \nabla_i f(x^t)||_{\infty} = ||\hat{\nabla} f(x^t) - \nabla f(x^t)||_{\infty} \), we need to calculate \( ||\hat{\nabla}_i f(x^t) - \nabla_i f(x^t)|| \) for a random \( i \in [d] \). Recall that \( \nabla_i f(x^t) = a_i^\top (\frac{1}{n} Ax^t) - a_i^\top y := a_i^\top q^t - a_i^\top y \), where \( q^t \) is the only part that cannot be updated by each user independently based solely
on local features. However, it can be iteratively updated based on the update rule of $x^t$, as

$$q^t := \frac{1}{n} A x^t = (1 - \gamma^{t-1}) \frac{1}{n} A x^{t-1} + \gamma^{t-1} \sum_{k=1}^{K} -\text{sign}(\hat{r}_k^{t-1}) \frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1}$$

$$= (1 - \gamma^{t-1}) q^{t-1} + \gamma^{t-1} \sum_{k=1}^{K} -\text{sign}(\hat{r}_k^{t-1}) \frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1}$$

$$= \sum_{\tau=0}^{t-1} \left( \sum_{k=1}^{K} -\text{sign}(\hat{r}_k^{t-1}) \frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1} \right) \gamma^\tau \prod_{j=\tau+1}^{t-1} (1 - \gamma^j).$$

The private counterpart is given by

$$q^t = (1 - \gamma^{t-1}) q^{t-1} + \gamma^{t-1} \sum_{k=1}^{K} -\text{sign}(\hat{r}_k^{t-1})(J(\frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1}) + \xi_k^t)$$

$$= \sum_{\tau=0}^{t-1} \left( \sum_{k=1}^{K} -\text{sign}(\hat{r}_k^{t-1})(J(\frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1}) + \xi_k^t) \right) \gamma^\tau \prod_{j=\tau+1}^{t-1} (1 - \gamma^j).$$

Noting that $\hat{\nabla}_i f(x^t) = (Ja_i)^\top q^t - a_i^\top y$, we need to bound

$$|\nabla_i f(x^t) - \hat{\nabla}_i f(x^t)| = |a_i^\top q^t - (Ja_i)^\top \hat{q}^t|,$$

which is upper bounded by a weighted summation of

$$|a_i^\top \left( \frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1} - (Ja_i)^\top (J(\frac{1}{n} a_{\hat{r}_k^{t-1}}^{t-1}) + \xi_k^t) \right) | \leq \frac{1}{n} |a_i^\top |a_{\hat{r}_k^{t-1}}^{t-1} - (Ja_i)^\top (Ja_{\hat{r}_k^{t-1}}^{t-1})| + |Ja_i^\top \xi_k^t|,$$

where the first term, the difference between the inner product of the original vectors and JL-transformed ones with JL parameter $\iota$, can be bounded by JL-Lemma and gives

$$\frac{1}{n} |a_i^\top a_{\hat{r}_k^{t-1}}^{t-1} - (Ja_i)^\top (Ja_{\hat{r}_k^{t-1}}^{t-1})| \leq \frac{1}{n} |a_i||a_{\hat{r}_k^{t-1}}||a_{\hat{r}_k^{t-1}}| \leq \frac{\text{nnz}(a)}{n};$$

while the second term can be bounded by Hölder inequality and the fact that $\xi_k^t : m$ with each element sampled from $\text{pert}' \sim \mathcal{N}(0, \frac{\sigma(J) \sqrt{KT} \sqrt{2(\log(\frac{1}{\delta}) + \epsilon/2)}}{n\epsilon/2})$,

$$|(Ja_i)^\top \xi| \leq ||Ja_i||_2||\xi||_2 \leq \sigma(J) \sqrt{\text{nnz}(a)} \sqrt{m} \sqrt{KT} \sigma(J) \sqrt{2(\log(\frac{1}{\delta}) + \epsilon/2)} \frac{n}{n\epsilon/2}.$$
With the obtained upper bound of a single term, by substituting \( \gamma^t = \frac{2}{p_{\min} + 2} \) in (D.3.33) and considering all \( K \) terms, the inexactness of the gradient is

\[
|\nabla_i f(x^t) - \hat{\nabla}_i f(x^t)| \leq \frac{K \text{nnz}(\mathbf{a})}{p_{\min} r_{\min} n} + K^{3/2} \sqrt{m} \sqrt{\text{nnz}(\mathbf{a})} \sigma(J)^2 \sqrt{2(\log(\frac{1}{\delta}) + \epsilon/2)} \frac{\sqrt{T}}{p_{\min} n^{-2}}.
\]

(D.3.37)

Hence, with \( D_M = 2\eta \) and \(|T^i| = K\tau\), the inexact gradient parameter should satisfy

\[
\theta_g (2\eta + K\tau) \gamma^t C_f E(K, \tau)^T \leq O\left( \frac{((K \text{nnz}(\mathbf{a}) + K^{3/2} \sqrt{\text{nnz}(\mathbf{a})} m \sigma(J)^2) / p_{\min}(2\eta + K\tau)) \sqrt{T} \sqrt{(\log(1/\delta) + \epsilon)}}{n\epsilon C_f E(K, \tau)^T} \right),
\]

(D.3.38)

which further gives

\[
\theta_g = O\left( \frac{((K \text{nnz}(\mathbf{a}) + K^{3/2} \sqrt{\text{nnz}(\mathbf{a})} m \sigma(J)^2) / p_{\min}(2\eta + K\tau)) \sqrt{T} \sqrt{(\log(1/\delta) + \epsilon)}}{n\epsilon C_f E(K, \tau)^T} \right);
\]

(D.3.39)

• Inexact linear oracle: Recall that \( \text{pert}_k \) is sampled from Laplace distribution with parameter \( (G_k \|M_k\|_1/n) \sqrt{2T \log(1/\delta/2)} / \epsilon_k / 4 \) for each user \( k \). Note that \( \epsilon_k = \epsilon \cdot \frac{G_k \|M_k\|_1}{\sum_{k=1}^KG_k \|M_k\|_1} \), which gives \( \text{pert}_k \sim \text{Lap}(\frac{\sum_{k=1}^KG_k \|M_k\|_1}{n\epsilon} \sqrt{32T \log(1/\delta/2)} \). By the concentration bound of the Laplace noise, and replacing \( \sum_{k=1}^KG_k \|M_k\|_1 \) with \( G'\|M\|_1 \) for notational simplicity due to \( \sum_{k=1}^KG_k \|M_k\|_1 \leq G\|M\|_1 = 2G\eta \), we have for any node \( k \), with probability at least \( 1 - \zeta \),

\[
\frac{\theta_g K\tau \gamma^t C_f E(K, \tau)^T}{2} \leq O\left( \frac{2G\eta \sqrt{32T \log(1/\delta/2) \log(K\tau T/\zeta)/(K\tau)}}{n\epsilon} \right).
\]

(D.3.40)

Let \( \zeta = 1/(2G\eta) \) and note that for any \( s, s' \in \mathcal{M} \) and any \( \nabla f(x) \), the perturbation \(|\langle s - s', \nabla f(x) \rangle| \leq 2G\eta\), then

\[
O\left( \frac{2G\eta \sqrt{32T \log(1/\delta/2) \log(K\tau T/\zeta)/(K\tau)}}{n\epsilon} \right)
\]

\[
\leq (1 - \zeta) \cdot O\left( \frac{2G\eta \sqrt{32T \log(1/(\delta/2) \log(2G\eta K\tau T)/(K\tau))}}{n\epsilon C_f E(K, \tau)^T} \right) + \zeta \cdot 2G\eta
\]

\[
= O\left( \frac{2G\eta \sqrt{32T \log(1/(\delta/2) \log(2G\eta K\tau T)/(K\tau))}}{n\epsilon} \right).
\]

(D.3.41)
Thus, the inexact linear oracle has the parameter,

$$
\varrho_l = O\left(\frac{2G\eta\sqrt{32T}\log(1/(\delta/2) \log(2G\eta K^\tau T)/(K^\tau)}{n\epsilon C_{f}^{E(K,\tau)} T}\right). \quad (D.3.42)
$$

Finally, we can show the utility guarantee. For notational simplicity, denote

$$
C_g = ((K^\tau \text{nnz}(a) + K^{3/2} \sqrt{\text{nnz}(a) \sigma(J)^2})/p_{\text{min}})\sqrt{\log(1/\delta) + \epsilon};
$$

$$
C_l = 2G\eta\sqrt{32\log(1/(\delta/2)}.
$$

By the convergence result,

$$
\mathbb{E}[f(x^T)] - \min_{x \in \mathcal{M}} f(x) = O\left(\frac{C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T} + \frac{\varrho C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T}\right)
$$

$$
= O\left(\frac{C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T} + \frac{[(2\eta + K^\tau)\varrho_l + K^\tau \varrho_l] C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T n \epsilon}\right)
$$

$$
= O\left(\frac{C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T} + \frac{\sqrt{T}(C_g \sqrt{m} + C_l \log(2G\eta K^\tau T)) C_{f}^{E(K,\tau)}}{p_{\text{min}}^2 T n \epsilon}\right).
$$

(D.3.43)

Substitute $T = \left(\frac{C_{f}^{E(K,\tau)} n \epsilon}{C_g + C_l}\right)^{\frac{1}{4}}$ (hence $\sqrt{m} = O(\log(n^{1/3}))$) in,

$$
\mathbb{E}[f(x^T)] - \min_{x \in \mathcal{M}} f(x) = O\left(\frac{(C_g + C_l)^{\frac{1}{2}} (C_{f}^{E(K,\tau)})^{\frac{1}{2}} \log(2G\eta K^\tau n)}{p_{\text{min}}^2 (n \epsilon)^{\frac{1}{2}}}\right)
$$

$$
= O\left(\frac{(K^\tau)^{\frac{1}{2}} (C_{f}^{E(K,\tau)})^{\frac{1}{2}} \log(2G\eta K^\tau n)}{(K^\tau)^{2} (n \epsilon)^{2}}\right).
$$

(D.3.44)

which is the utility guarantee.
Appendix E

Proofs of Materials in Chapter 8

E.1 Additional Materials to Section 8.3.1

E.1.1 Window Tree Mechanism with Gamma Noise Perturbation

Proof of Lemma 8.3.1

Proof. Denote the input streaming sequence by $\Gamma_t = (v_1, v_2, \ldots, v_t)$ and denote the $\ell_2$ sensitivity of $v$ by $\Delta_2 = \max_{v, v'} \|v - v'\|_2$. The differential privacy with window size $W$ only considers privacy preserving for the latest $W$ entities, i.e. $(v_{t-W+1}, \ldots, v_t)$. For any $v_w$ changes, it only affects at most $\lceil \log_2 W \rceil + 1$ nodes. Note that by proposition A1, with $\epsilon = \epsilon/(\lceil \log_2 W \rceil + 1)$ and $c = \frac{\Delta_2}{\epsilon/(\lceil \log_2 W \rceil + 1)}$, each node is $\epsilon/(\lceil \log_2 W \rceil + 1)$-differentially private. Then, by simple composition theorem, the window tree mechanism is $\epsilon$-differentially private. \hfill \Box

Proof of Lemma 8.3.2

Proof. Note that there are at most $(\lceil \log_2 W \rceil + 1)$ noise vectors. Hence, $\|s_t - \sum_{i=1}^{t} v_i\|_2 \leq \|\sum_{i=1}^{\lceil \log_2 W \rceil + 1} n_i\|_2 \leq \sqrt{(\lceil \log_2 W \rceil + 1)}\|n_t\|_2$. Then, by proposition A2 with $k = d$ and $\theta = \frac{\Delta_2(\lceil \log_2 W \rceil + 1)}{\epsilon}$ there, we have $\|n_t\|_2 \leq \frac{d\Delta_2(\lceil \log_2 W \rceil + 1) \ln \frac{2}{\epsilon}}{\epsilon}$ with
probability at least $1 - \beta$. As a result, with probability at least $1 - \beta$, where we have:

$$\|s_t - \sum_{i=1}^{t} v_i\|_2 \leq \left( \frac{d \Delta_2 \log_2^{1.5} W \ln \frac{d}{\beta}}{\epsilon} \right). \quad (E.1.1)$$

\[\square\]

### E.1.2 Window Tree Mechanism with Gaussian Noise Perturbation

#### Proof of Lemma 8.3.3

**Proof.** Denote the input streaming sequence by $\Gamma_t = (v_1, v_2, ..., v_t)$, and denote the $\ell_2$ sensitivity of $v$ by $\Delta = \max_{v,v'} ||v - v'||_2$. The differential privacy with window size $W$ only considers privacy preserving for the latest $W$ entities, i.e. $(v_{t-W+1}, ..., v_t)$. For any $v_w$ changes, it only affects $(\lceil \log_2 W \rceil + 1)$ nodes. Note that by proposition A3, with $\epsilon = \frac{\epsilon}{\sqrt{2(\lceil \log_2 W \rceil + 1)) \ln \frac{2}{\delta}}} \quad \text{and}$

$$c^2 = 2 \cdot 2^{(\lceil \log_2 W \rceil + 1) \cdot \ln \frac{2}{\delta}} \geq 2(\ln 2(\lceil \log_2 W \rceil + 1)) \cdot (\ln \frac{2}{\delta})$$

$$\geq 2\ln 2((\lceil \log_2 W \rceil + 1) + \ln \frac{2}{\delta}) = 2 \ln[2(\lceil \log_2 W \rceil + 1) \cdot \frac{2}{\delta}] \quad (E.1.2)$$

$$\geq 2 \ln \frac{1.25}{\frac{2}{\delta}}$$

each node preserves $(\epsilon', \delta_1)$-DP, where $\epsilon' = \frac{\epsilon}{\sqrt{2(\lceil \log_2 W \rceil + 1)) \ln \frac{2}{\delta}}}$, $\delta_1 = \frac{\delta}{2(\lceil \log_2 W \rceil + 1))}. \quad \text{Then by strong composition theorem, with } \delta_2 = \frac{\delta}{2}, \text{ the algorithm is } (\sqrt{2(\lceil \log_2 W \rceil + 1)) \ln \frac{2}{\delta}}, \epsilon', (\lceil \log_2 W \rceil + 1)) \cdot \delta_1 + \delta_2)$-DP, i.e. $(\epsilon, \delta)$-DP.

\[\square\]

#### Proof of Lemma 8.3.4

**Proof.** Note that $s_t$ is the summation of $\sum_{i=1}^{t} v_i$ with at most $(\lceil \log_2 W \rceil + 1)$ $d$-dimensional Gaussian noise vectors. Hence, similarly, by proposition A4, we have
with probability at least $1 - \beta$,

$$
||s_t - \sum_{i=1}^{t} v_i||_2 = O(\sqrt{(|\log_2 W| + 1)||n_t||_2^2})
= O(\sqrt{(|\log_2 W| + 1)\sqrt{\frac{d}{\beta}}\Delta_2 \log_2 W \ln \frac{2}{\delta}/\epsilon}) \quad (E.1.3)
= O\left(\frac{\sqrt{d\Delta_2 \log_2^1 W \ln \frac{1}{\beta}}}{\epsilon}\right).
$$

\[\square\]

### E.2 Additional Materials to Section 8.3.2

This part contains the proof of theorems in subsection 8.3.2. We begin with defining the following notation.

$$
x_{t+1}^* = \arg \min_{x \in P} J_t(x) = \arg \min_{x \in P} \left(\sum_{\tau=1}^{t} \langle \nabla f_{\tau}(\hat{x}_\tau), x \rangle + \frac{\mu}{2}||x - \hat{x}_\tau||_2^2\right), \quad (E.2.4)
$$

$$
\hat{J}_t(x) = \langle \hat{g}_t, x \rangle + \sum_{\tau=1}^{t} \frac{\mu}{2}||x - \hat{x}_\tau||_2^2 + C_0 \frac{\mu}{2}||x - x_1||_2^2 = J(x) + \langle n, x \rangle, \quad (E.2.5)
$$

where $n$ denotes the sum of noise vectors added to the relevant nodes of the window tree mechanism. Further, we define

$$
\hat{x}_{t+1} = \arg \min_{x \in P} \hat{J}_t(x). \quad (E.2.6)
$$

$$
\tilde{f}_t(x) = f_t(\hat{x}_t) + \langle \nabla f_t(\hat{x}_t), x - \hat{x}_t \rangle + \frac{\mu}{2}||x - \hat{x}_t||_2^2. \quad (E.2.7)
$$

We denote by $x^*$ and $\tilde{x}^*$ as

$$
x^* = \arg \min_{x \in P} \sum_{t=1}^{T} f_t(x), \quad \tilde{x}^* = \arg \min_{x \in P} \sum_{t=1}^{T} \tilde{f}_t(x). \quad (E.2.8)
$$

The following lemma will be used in both proofs of the current and next subsection.

**Lemma E.2.1.** For any $t$, the following inequality holds,

$$
\sum_{\tau=1}^{t} (f_{\tau}(\hat{x}_\tau) - f_{\tau}(x^*)) \leq \sum_{\tau=1}^{t} (\hat{f}_{\tau}(\hat{x}_\tau) - \tilde{f}_{\tau}(\tilde{x}^*)). \quad (E.2.9)
$$
Proof. By definition of $\tilde{f}_\tau$, $f_\tau(x) = \tilde{f}_\tau(x)$ and with the strongly convexity of $f_\tau(x)$, $\tilde{f}_\tau(x) \leq f_\tau(x)$, for any $x$ and $\tau = [t]$. Therefore $\sum_{\tau=1}^{t} \tilde{f}_\tau(x) \leq \sum_{\tau=1}^{t} f_\tau(x)$. Note that $\hat{x}^*$ is the optimum of $\sum_{\tau=1}^{t} \tilde{f}_\tau(x)$ while $x^*$ is the optimum of $\sum_{\tau=1}^{t} f_\tau(x)$, thus $\sum_{\tau=1}^{t} \tilde{f}_\tau(x^*) \leq \sum_{\tau=1}^{t} f_\tau(x^*)$. Combining this with $f_\tau(\hat{x}_\tau) = \tilde{f}_\tau(\hat{x}_\tau)$ for any $\tau$, we conclude that
\[
\sum_{\tau=1}^{t} (f_\tau(\hat{x}_\tau) - f_\tau(x^*)) \leq \sum_{\tau=1}^{t} (\tilde{f}_\tau(\hat{x}_\tau) - \tilde{f}_\tau(x^*)).
\] (E.2.10)

\[\] (E.2.10)

\section*{E.2.1 Proof of Theorem 8.3.7 and 8.3.8}

Proof.

\[
\text{Regret}(T) = \sum_{t=1}^{T} f_1(\hat{x}_t) - \arg \min_{x \in P} \sum_{t=1}^{T} f_1(x) = \sum_{t=1}^{T} (f_1(\hat{x}_t) - f_1(x^*)) \\
\leq \sum_{t=1}^{T} (\tilde{f}_1(\hat{x}_t) - \tilde{f}_1(x^*)) \leq \sum_{t=1}^{T} (\tilde{f}_1(x^*_t) - \tilde{f}_1(x^*)) + \frac{2(L + \mu D)||n||_2 \ln T}{\mu} \\
\leq \frac{2(L + \mu D)^2 \ln T}{\mu} + \frac{2(L + \mu D)||n||_2 \ln T}{\mu},
\] (E.2.11)

where (i) is by Lemma E.2.1, (ii) is by Lemma 8 in [130] and (iii) is by Fact 9 and Claim 10 in [130], which are originated from [50]. Substitute the high probability upper bound for $||n||_2$, we get

\[
\text{Regret}(T) \leq O\left(\frac{2(L + \mu D)^2 \ln T}{\mu} + \frac{d(L + \mu D)^2 \log_2 W \ln \frac{d}{\beta} \ln T}{\mu}\right)
\leq O\left(\frac{d(L + \mu D)^2 \log_2^{1.5} W \ln \frac{d}{\beta} \ln T}{\mu}\right),
\] (E.2.12)

for window $\epsilon$-differential privacy with window size $W$ based on Gamma perturbation, which is the result of Theorem 8.3.7. Subsequently, we have:

\[
\text{Regret}(T) \leq O\left(\frac{2(L + \mu D)^2 \ln T}{\mu} + \frac{\sqrt{d}(L + \mu D)^2 \log_2 W \ln \frac{1}{\delta^{0.5}} \ln T}{\mu \epsilon}\right)
\leq O\left(\frac{\sqrt{d}(L + \mu D)^2 \log_2^{1.5} W \ln \frac{1}{\delta^{0.5}} \ln T}{\mu \epsilon}\right),
\] (E.2.13)

for window $(\epsilon, \delta)$-differential privacy with window size $W$ based on Gaussian perturbation, which is the result of Theorem 8.3.8. □
E.3 Additional Materials to Section 8.3.3

This part mainly contains the proof of Theorem 8.3.10 and 8.3.11, i.e. the regret analysis of WDP-OCOL. The privacy analysis of WDP-COCOL is the still based on the post-processing argument, thus we omit it here.

We first introduce the following notation, some of which are the same as in the WDP-COCOP. We denote by $x_{t+1}^*$ the minimum of the function $J(x)$, by $\hat{x}_{t+1}$ the output of the algorithm at the $t$-th iteration, i.e. after data sample $t$ coming in.

$x_{t+1}^* = \arg \min_{x \in \mathcal{P}} J_t(x) = \arg \min_{x \in \mathcal{P}} \left( \sum_{\tau=1}^{t} \langle \nabla f_{\tau}(\hat{x}_\tau), x \rangle + \frac{\mu}{2}||x - \hat{x}_\tau||_2^2 + C_0 \frac{\mu}{2} ||x - x_1||_2^2 \right)$, \hspace{1cm} (E.3.14)

$\hat{J}_t(x) = (\langle g_t, x \rangle + \sum_{\tau=1}^{t} \frac{\mu}{2}||x - \hat{x}_\tau||_2^2 + C_0 \frac{\mu}{2} ||x - x_1||_2^2 = J(x) + \langle n, x \rangle), \hspace{1cm} (E.3.15)$

$\hat{x}_{t+1} = \arg \min_{x \in \mathcal{P}} \hat{J}_t(x)$, \hspace{1cm} (E.3.16)

$\tilde{f}_t(x) = f_t(\hat{x}_t) + \langle \nabla f_t(\tilde{x}_t), x - \hat{x}_t \rangle + \frac{\mu}{2}||x - \hat{x}_t||_2^2$. \hspace{1cm} (E.3.17)

We denote by $x^*$ and $\tilde{x}^*$ as

$x^* = \arg \min_{x \in \mathcal{P}} \sum_{t=1}^{T} f_t(x), \hspace{1cm} \tilde{x}^* = \arg \min_{x \in \mathcal{P}} \sum_{t=1}^{T} \tilde{f}_t(x)$. \hspace{1cm} (E.3.18)

Again, we recall that $\tilde{f}_t(x)$ is $(L + \mu D)$-Lipschitz continuous and $J_t(x)$ is $\mu(t + C_0)$-strongly convex. We also need Lemma [E.2.1] in the following proof.

E.3.1 Main Proof of Theorem 8.3.10 and 8.3.11

Proof. The big picture of the regret analysis is as follows:

$\text{Regret}(T) = \sum_{t=1}^{T} f_t(\hat{x}_t) - \arg \min_{x \in \mathcal{P}} \sum_{t=1}^{T} f_t(x) = \sum_{t=1}^{T} (f_t(\hat{x}_t) - f_t(x^*))$ \hspace{1cm}

$(i) \leq \sum_{t=1}^{T} (\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x^*)) = \left[ \sum_{t=1}^{T} (\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x^*_t)) \right] + \left[ \sum_{t=1}^{T} (\tilde{f}_t(x^*_t) - \tilde{f}_t(\tilde{x}^*)) \right] \hspace{1cm}$ \hspace{1cm} (E.3.19)

$(ii)$
The inequality \((i)\) is according to Lemma \([E.2.1]\). Next, we get the upper bound terms in bracket \((ii)\) and \((iii)\). The upper bound of \((iii)\) is by Lemma 6.6 from \([42]\) (they give a proof in their Appendix A and can also be found in \([67]\)), which ensures that
\[
\sum_{t=0}^{T} (\hat{f}_t(x_{t+1}^*) - \tilde{f}_t(\tilde{x}^*)) = \sum_{t=1}^{T} (\hat{f}_t(x_{t+1}^*) - \tilde{f}_t(x^*)) + C_0 \mu \frac{1}{2} (||x_1^* - x_1||_2 - ||\tilde{x}^* - x_1||_2) \leq 0. 
\]  
(E.3.20)

Hence,
\[
\sum_{t=1}^{T} (\hat{f}_t(x_{t+1}^*) - \tilde{f}_t(\tilde{x}^*)) \leq \frac{C_0 \mu D^2}{2} = \frac{(25\rho^2)^2 \mu D^2}{2}.
\]  
(E.3.21)

We prove in Lemma \([E.3.1]\) and Lemma \([E.3.2]\) that \((ii)\) can be upper bounded as
\[
\sum_{t=1}^{T} (\hat{f}_t(\tilde{x}_t) - \tilde{f}_t(x_{t+1}^*)) \leq \frac{28\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{2(L + \mu D) \ln T}{\mu} ||n||_2, 
\]  
(E.3.22)

where \(n\) is the summation of random noise vectors added to relevant nodes in the window tree mechanism. The proof of Lemma \([E.3.1]\) is by induction and is the main contribution of adapting the nonprivate online conditional gradient method \([42]\) to our differential privacy setting.

With the upper bounds of \((ii)\) and \((iii)\), we have the regret bound as,
\[
\text{Regret}(T) = \sum_{t=1}^{T} f_t(\hat{x}_t) - \arg \min_{x \in \mathbb{R}} \sum_{t=1}^{T} f_t(x) = \sum_{t=1}^{T} (\hat{f}_t(\hat{x}_t) - \hat{f}_t(x^*)) \\
\leq \sum_{t=1}^{T} (\tilde{f}_t(\tilde{x}_t) - \tilde{f}_t(\tilde{x}^*)) = \sum_{t=1}^{T} (\hat{f}_t(\tilde{x}_t) - \tilde{f}_t(x_{t+1}^*)) + \sum_{t=1}^{T} (\tilde{f}_t(x_{t+1}^*) - \hat{f}_t(\tilde{x}^*)) \\
\leq \frac{28\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{2(L + \mu D) \ln T}{\mu} ||n||_2 + \frac{(25\rho^2)^2 \mu D^2}{2} \\
= O\left(\frac{\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{(L + \mu D) \ln T}{\mu} ||n||_2\right). 
\]  
(E.3.23)

Then, by the estimation of \(||n||_2\), we have the regret bound with Gamma perturbation with window size \(W\) as,
\[
\text{Regret}(T) = O\left(\frac{\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{d(L + \mu D)^2 \log^{1.5} W \ln \frac{\delta}{\beta} \ln T}{\mu \epsilon}\right), 
\]  
(E.3.24)
which is the result of Theorem 8.3.10. Also, with Gaussian perturbation, we have the regret bound with Gaussian perturbation with window size $W$ as,

$$Regret(T) = O\left(\frac{\mu^2 (L + \mu D)^2 \ln T}{\mu} + \sqrt{d} (L + \mu D)^2 \log^2 1.5 \ln \frac{1}{\beta} \ln \frac{1}{\beta} \ln T\right),$$

(E.3.25)

which is the result of Theorem 8.3.11. Finally, by substituting in a particular setting of $\rho$ for general polytope $\mathcal{P}$ to be $\rho = c \sqrt{d}$, where $c$ is a constant related to the geometry of the polytope $\mathcal{P}$. The regret would be

$$Regret(T) = O\left(\frac{d (L + \mu D)^2 \log^2 1.5 W \ln \frac{d}{\beta} \ln T}{\mu \epsilon}\right),$$

(E.3.26)

for $\epsilon$ differential privacy with window size $W$ based on Gamma perturbation,

$$Regret(T) = O\left(\max(\sqrt{d} \epsilon, \log^2 1.5 W \ln \frac{1}{\delta} \ln \frac{1}{\beta}) \cdot \sqrt{d} (L + \mu D)^2 \ln T\right),$$

(E.3.27)

for $(\epsilon, \delta)$-differential privacy with window size $W$ based on Gaussian perturbation.

Lemma E.3.1. Let $r_t = \sqrt{\frac{\eta_t}{\mu (t + C_0)}} + \frac{2 (L + \mu D)}{\mu (t + C_0)}$, $\eta_t = \frac{C_0^2}{\mu (t + C_0)}$. For any $t \in [T]$, it holds that $J_{t-1}(\hat{x}_t) - J_{t-1}(x^*_t) \leq \eta_t$ and $||\hat{x}_t - x^*_t||_2 \leq \sqrt{\frac{\eta_t}{\mu (t + C_0)}}$.

Proof.

$$||\hat{x}_t - x^*_{t+1}||_2 = ||\hat{x}_t - x^*_t + x^*_t - x^*_{t+1}||_2$$

$$\leq ||\hat{x}_t - x^*_t||_2 + ||x^*_t - x^*_{t+1}||_2$$

$$\stackrel{(i)}{\leq} \sqrt{\frac{2 \eta_t}{\mu (t - 1 + C_0)}} + ||x^*_t - x^*_{t+1}||_2$$

$$\stackrel{(ii)}{\leq} \sqrt{\frac{2 \eta_t}{\mu (t - 1 + C_0)}} + \frac{2 (L + \mu D)}{\mu (t + C_0)}$$

(E.3.28)

$$\leq \sqrt{\frac{4 \eta_t}{\mu (t + C_0)}} + \frac{2 (L + \mu D)}{\mu (t + C_0)}$$

$$= r_t,$$
where inequality (i) is by induction and (ii) is by the following,
\[
\frac{1}{2} \|x_t^* - x_{t+1}^*\|^2 \leq \frac{J_t(x_t^*) - J_t(x_{t+1}^*)}{\mu(t + C_0)}
\]
\[
= \frac{J_{t-1}(x_t^*) - J_{t-1}(x_{t+1}^*) + \tilde{f}_t(x_t^*) - \tilde{f}_t(x_{t+1}^*)}{\mu(t + C_0)}
\]
\[
\leq 0 + (L + \mu D)\|x_t^* - x_{t+1}^*\|_2
\]
Therefore,
\[
\|x_t^* - x_{t+1}^*\|_2 \leq \frac{2(L + \mu D)}{\mu(t + C_0)}.
\]
Also, by the Lipschitz continuous of \(\tilde{f}_t(x)\), we have
\[
|\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_{t+1}^*)| \leq (L + \mu D)\|\hat{x}_t - x_{t+1}^*\|_2 \leq (L + \mu D)r_t.
\]
By the assumption of the \(\mathcal{LLOO}\) operator and denote the sum of noise vectors added to the relevant nodes in tree mechanism by \(n\), we have
\[
\langle \nabla J_t(\hat{x}_t), p_t - \hat{x}_t \rangle = \langle \nabla \tilde{J}_t(\hat{x}_t) - n, p_t - \hat{x}_t \rangle
\]
\[
= \langle \nabla \tilde{J}_t(\hat{x}_t), p_t - \hat{x}_t \rangle - \langle n, p_t - \hat{x}_t \rangle
\]
\[
\leq \langle \nabla \tilde{J}_t(\hat{x}_t), p_t - \hat{x}_t \rangle + \|n\|_2\|p_t - \hat{x}_t\|_2
\]
\[
\leq \langle \nabla \tilde{J}_t(\hat{x}_t), x_{t+1}^* - \hat{x}_t \rangle + \|n\|_2\|p_t - \hat{x}_t\|_2
\]
\[
= \langle \nabla J_t(x_{t+1}^*) + n, x_{t+1}^* - \hat{x}_t \rangle + \|n\|_2\|p_t - \hat{x}_t\|_2
\]
\[
\leq J_t(x_{t+1}^*) - J_t(\hat{x}_t) + \|n\|_2\|\hat{x}_t - x_{t+1}^*\|_2 + \|p_t - \hat{x}_t\|_2
\]
\[
\leq J_t(x_{t+1}^*) - J_t(\hat{x}_t) + 2r_t\|n\|_2.
\]
With the above relation, we further have
\[
J_t(\hat{x}_{t+1}) - J_t(x_{t+1}^*) \leq J_t(\hat{x}_t) + \alpha \langle \nabla J_t(\hat{x}_t), p_t - \hat{x}_t \rangle + \frac{\alpha^2 \mu(t + C_0)}{2}\|p_t - \hat{x}_t\|_2^2 - J_t(x_{t+1}^*)
\]
\[
\leq J_t(\hat{x}_t) + \alpha (J_t(x_{t+1}^*) - J_t(\hat{x}_t) + 2r_t\|n\|_2) + \frac{\alpha^2 \mu(t + C_0)}{2}\|p_t - \hat{x}_t\|_2^2 - J_t(x_{t+1}^*)
\]
\[
\leq (1 - \alpha)(J_t(\hat{x}_t) - J_t(x_{t+1}^*)) + 2\alpha\|n\|_2r_t + \frac{\mu \rho^2 \alpha^2}{2}(t + C_0)r_t^2.
\]
To bound \(J_t(\hat{x}_t) - J_t(x_{t+1}^*)\), we have
\[
J_t(\hat{x}_t) - J_t(x_{t+1}^*) = J_{t-1}(\hat{x}_t) - J_{t-1}(x_{t+1}^*) + \tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_{t+1}^*)
\]
\[
\leq J_{t-1}(\hat{x}_t) - J_{t-1}(x_{t+1}^*) + \tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_{t+1}^*)
\]
\[
\leq \eta_t + (L + \mu D)r_t.
\]
\[
J_t(\hat{x}_{t+1}) - J_t(x^*_t) \leq (1 - \alpha)(\eta_t + (L + \mu D)r_t) + 2\alpha ||n||_2r_t + \frac{\mu^2 \alpha^2}{2}(t + C_0)r_t^2
\]

\[
= (1 - \alpha)\eta_t + ((1 - \alpha)(L + \mu D) + 2\alpha ||n||_2)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})
\]

\[
+ \frac{\mu^2 \alpha^2}{2}(t + C_0)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})^2.
\]

(E.3.35)

By \((a + b)^2 \leq 2a^2 + 2b^2\) for \(r_t^2\),

\[
J_t(\hat{x}_{t+1}) - J_t(x^*_t) \leq (1 - \alpha)\eta_t + ((1 - \alpha)(L + \mu D) + 2\alpha ||n||_2)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})
\]

\[
+ \frac{\mu^2 \alpha^2}{2}(t + C_0)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})^2
\]

\[
= (1 - \alpha)\eta_t + ((1 - \alpha)(L + \mu D) + 2\alpha ||n||_2)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})
\]

\[
+ (4\rho^2 \alpha^2)(\eta_t + \frac{(L + \mu D)^2}{\mu(t + C_0)})
\]

\[
= (1 - \alpha)\eta_t + (1 - \alpha)(L + \mu D)(1 + \frac{2\alpha ||n||_2}{(1 - \alpha)(L + \mu D)})(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}} + \frac{2(L + \mu D)}{\mu(t + C_0)})
\]

\[
+ (4\rho^2 \alpha^2)(\eta_t + \frac{(L + \mu D)^2}{\mu(t + C_0)})
\]

\[
= (1 - \alpha)(\eta_t + (1 + C_W)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}}(L + \mu D) + \frac{2(L + \mu D)^2}{\mu(t + C_0)}))
\]

\[
+ (4\rho^2 \alpha^2)(\eta_t + \frac{(L + \mu D)^2}{\mu(t + C_0)})
\]

\[
\leq (1 - \alpha + 4\rho^2 \alpha^2)(\eta_t + (1 + C_W)(\sqrt{\frac{4\eta_t}{\mu(t + C_0)}}(L + \mu D) + \frac{2(L + \mu D)^2}{\mu(t + C_0)})),
\]

(E.3.36)

where

\[
C_W = \frac{2\alpha ||n||_2}{(1 - \alpha)(L + \mu D)}.
\]

(E.3.37)
Then, we substitute \( \eta_t = \frac{C_t^2}{\mu(t+C_0)} \) and \( \alpha = \frac{1}{5\rho^2} \) in eq. (E.3.36),

\[
J_t(\hat{x}_{t+1}) - J_t(x_{t+1}^*) = (1 - \frac{1}{25\rho^2})(\frac{C_t^2}{\mu(t+C_0)}) + (1 + C_W)(\frac{2C_1}{\mu(t+C_0)}(L + \mu D) + \frac{2(L + \mu D)^2}{\mu(t+C_0)})
\]
\[
= (1 - \frac{1}{25\rho^2})(\frac{C_t^2 + 2(1 + C_W)(L + \mu D)C_1 + 2(L + \mu D)^2}{\mu(t+C_0)})
\]
\[
= (1 - \frac{1}{25\rho^2})(1 + 2(1 + C_W)(L + \mu D)(\frac{1}{C_1} + 2(L + \mu D)^2) \frac{C_t^2}{\mu(t+C_0)}).
\]

(E.3.38)

By our choice of \( C_1 \), i.e. as long as \( C_1 \geq (1 + 25\rho^2)(1 + C_W)(L + \mu D) \) we have

\[
2(1 + C_W)(L + \mu D)(\frac{1}{C_1} + 2(L + \mu D)^2) \frac{1}{C_t^2} \leq \frac{1}{25\rho^2}.
\]

(E.3.39)

Hence,

\[
J_t(\hat{x}_{t+1}) - J_t(x_{t+1}^*) \leq (1 - \frac{1}{(25\rho^2)^2}) \frac{C_t^2}{\mu(t+C_0)}.
\]

(E.3.40)

Let \( C_0 = (25\rho^2)^2 \), we have

\[
J_t(\hat{x}_{t+1}) - J_t(x_{t+1}^*) \leq (1 - \frac{1}{C_0}) \frac{C_t^2}{\mu(t+C_0)} < (1 - \frac{1}{t+1+C_0}) \frac{C_t^2}{\mu(t+C_0)}
\]
\[
= \frac{C_t^2}{\mu(t+1+C_0)} = \eta_{t+1}.
\]

This completes our induction.

\[\square\]

Lemma E.3.2. (ii)-term in the proof of Theorem 7 has the following bound,

\[
\sum_{t=1}^{T} (\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_{t+1}^*)) \leq \frac{28\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{2(L + \mu D) \ln T}{\mu} ||n||_2.
\]

(E.3.42)

Proof. We begin with \( ||\hat{x}_t - x_{t+1}^*||_2 \leq ||x_t^* - x_{t+1}^*||_2 + ||\hat{x}_t - x_t^*||_2 \). Then, once we bound this term, we can obtain the bound for \( (\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_t^*)) \) by the Lipschitz continuity, i.e. \( (\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x_t^*)) \leq (L + \mu D)||\hat{x}_t - x_{t+1}^*||_2 \).

By strong convexity of \( J_{t-1}(x) \),

\[
||\hat{x}_t - x_t^*||_2 \leq \sqrt{\frac{J_{t-1}(\hat{x}_t) - J_{t-1}(x_t^*)}{\mu(t-1+C_0)}} \leq \sqrt{\frac{\eta_t}{\mu(t-1+C_0)}}.
\]

(E.3.43)

We substitute \( \eta_t \) and \( C_0 \) in eq. (E.3.43),

\[
||\hat{x}_t - x_t^*||_2 \leq \sqrt{\frac{C_t^2}{\mu^2(t-1+(25\rho^2)^2)(t+(25\rho^2)^2)}} \leq \frac{C_t}{\mu t} = \frac{26\rho^2(L + \mu D)(1 + C_W)}{\mu t}
\]

(E.3.44)
Back to $||\hat{x}_t - x^*_t||_2$, with eq. (E.3.30) ($||x^*_t - x^*_{t+1}||_2 \leq \frac{2(L + \mu D)}{\mu (t + C_0)}$) in the proof of previous lemma,

$$
||\hat{x}_t - x^*_t||_2 \leq ||x^*_t - x^*_{t+1}||_2 + ||\hat{x}_t - x^*_t||_2 \leq \frac{2(L + \mu D)}{\mu t} + \frac{26\rho^2(L + \mu D)(1 + C_W)}{\mu t} \\
\leq \frac{28\rho^2(L + \mu D)}{\mu t} + \frac{26\rho^2(L + \mu D)}{\mu t} C_W \\
= \frac{28\rho^2(L + \mu D)}{\mu t} + \frac{2/5}{(1 - 1/(5\rho^2))\mu t} ||n||_2 \\
\leq \frac{28\rho^2(L + \mu D)}{\mu t} + \frac{2}{\mu t} ||n||_2.
$$

(E.3.45)

Again, by the Lipschitz continuous of $\tilde{f}_t(x)$,

$$
\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x^*_t) \leq (L + \mu D)||\hat{x}_t - x^*_t||_2 \leq \frac{28\rho^2(L + \mu D)^2}{\mu t} + \frac{2(L + \mu D)}{\mu t} ||n||_2.
$$

(E.3.46)

$$
\sum_{t=1}^{T}(\tilde{f}_t(\hat{x}_t) - \tilde{f}_t(x^*_t)) \leq \sum_{t=1}^{T}\left(\frac{28\rho^2(L + \mu D)^2}{\mu t} + \frac{2(L + \mu D)}{\mu t} ||n||_2\right) \\
\leq \frac{28\rho^2(L + \mu D)^2 \ln T}{\mu} + \frac{2(L + \mu D) \ln T}{\mu} ||n||_2.
$$

(E.3.47)
Appendix F

Publication List

F.1 Published Papers


5. Yiu-ming Cheung, Jian Lou, *Proximal Average Approximated Incremen-

### F.2 Submitted Papers


CURRICULUM VITAE

Academic qualification of the thesis author, Mr. LOU Jian:

• Received the degree of Bachelor of Science from the Zhejiang University, June 2013.

August 2018