The Dissertation Committee for Qi Lei
certifies that this is the approved version of the following dissertation:

Provably Effective Algorithms for Min-Max Optimization

Committee:

Inderjit S. Dhillon, Supervisor

Georgios-Alexandros Dimakis, Co-Supervisor

Per-Gunnar Martinsson

Tan Bui

George Biros

Qixing Huang

Provably Effective Algorithms for Min-Max Optimization

by

Qi Lei

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2020

Acknowledgments

I would like to give the most sincere thanks to my advisors Alexandros G. Dimakis and Inderjit S. Dhillon. Throughout my PhD, they constantly influence me with their splendid research taste, vision, and attitude. Their enthusiasm in the pursuit of challenging and fundamental machine learning problems has inspired me to do so as well. They encouraged me to continue the academic work and they have set as role models of being a professor with an open mind in discussions, criticisms, and collaborations.

I was fortunate in relatively big research groups throughout my stay at UT, where I was initially involved in Prof. Dhillon's group and later more in Prof. Dimakis' group. Both groups consist of good researchers and endow me with a collaborative research environment. The weekly group meetings push me to study different research directions and ideas on a daily basis. I am grateful to all my colleagues, especially my collaborators among them: Kai Zhong, Jiong Zhang, Ian Yen, Prof. Cho-Jui Hsieh, Hsiang-Fu Yu, Chao-Yuan Wu, Ajil Jalal, Jay Whang, Rashish Tandon, Shanshan Wu, Matt Jordan, Sriram Ravula, Erik Lindgren, Dave Van Veen, Si Si, Kai-Yang Chiang, Donghyuk Shin, David Inouye, Nikhil Rao, and Joyce Whang. I am also fortunate enough to collaborate with some other professors from UT like Prof. Constantine Caramanis, and Pradeep Ravikumar (now at CMU).

I also want to express my gratefulness to Professor Qixing Huang, Tan Bui, George Biros, Per-Gunnar Martinsson, together with my advisors to serve as my thesis committee. I am especially thankful for Prof. Qixing Huang's encouragement for me to pursue academic life.

I am thankful for all my other collaborators outside UT, where an unexhaustive list includes: Prof. Jason Lee, Jinfeng Yi, Lingfei Wu, Pin-Yu Chen, Arnaud Vandaele, Nicolas Gillis, Amir Gholami, Prof. Constantinos Daskalakis, Simon Du, Wei Hu, Nikos Karampatziakis, Prof. Sham Kakade, Prof. Kurt Keutzer, and Prof. Michael Mahoney. Without them, I could not have accomplished a decent amount of research that covers various sub-areas including numerical optimization, distributed learning, neural network architectures, deep learning theory, matrix analysis, and adversarial attacks/robustness. The machine learning/deep learning community is developing at an incredibly fast speed. No individual can follow all the work in all sub-areas. Without the collaborations with these wonderful researchers, I could not get a whole picture of what is fundamental to study in each area.

Among them I am especially thankful for Jinfeng Yi, the first mentor for my first internship. We have collaborated in four papers and two patents over that one internship. It was an enjoyable summer with rich intellectual gains. Jinfeng helped me develop my vision and understand many practical aspects of data mining and deep learning.

I am utmostly grateful to Prof. Jason Lee, who will also be my postdoc host. I could not have finished the last part of the thesis without his guidance. His insight in fundamental research for deep learning theory influenced me, guide and inspire me to work in this area.

Finally, my earnest gratitude is to my parents who supported me throughout my life. They are considerate, attentive, and supportive. They are good listeners in life and good role models in career. They treat me as friends and give me the most freedom, care, and unconditional love. They are the best parents one can hope for.

Provably Effective Algorithms for Min-Max Optimization

Publication No. _____

Qi Lei, Ph.D.
The University of Texas at Austin, 2020

Supervisors: Inderjit S. Dhillon
Georgios-Alexandros Dimakis

Many fundamental machine learning tasks can be formulated as min-max optimization. This motivates us to design effective and efficient first-order methods that provably converge to the global min-max points. For this purpose, this thesis focuses on designing practical algorithms for several specific machine learning tasks. We considered some different settings: unconstrained or constrained strongly-convex (strongly-)concave, constrained convex-concave, and nonconvex-concave problems. We tackle the following concrete questions by studying the above problems:

1. Can we reformulate a single minimization problem to two-player games to help reduce the computational complexity of finding global optimal points?

2. Can projection-free algorithms achieve last-iterate convergence for constrained min-max optimization problems with the convex-concave landscape?

3. Can we show that stochastic gradient descent-ascent, a method commonly used in practice for GAN training, actually finds global optima and can learn a target distribution?

We make progress on these questions by proposing practical algorithms with theoretical guarantees. We also present extensive empirical studies to verify the effectiveness of our proposed methods.

Table of Contents

List of Tables

List of Figures

Chapter 1

Overview

1.1 Overview of Min-Max Optimization

An important research area in machine learning involves multiple agents with different objectives interacting with each other. These problems can be mathematically modeled as n-player games. Therefore, as a starting point, this thesis focuses on two-player zero-sum games, i.e., **min-max optimization**. The formal mathematical formulation of this problem is:

$$\min_{\mathbf{x}\in\mathcal{X}} \max_{\mathbf{y}\in\mathcal{Y}} f(\mathbf{x}, \mathbf{y}). \tag{1.1}$$

The goal of the first player $\boldsymbol{x}$ is to minimize the function value $f(\boldsymbol{x}, \boldsymbol{y})$ within allowed actions in $\mathcal{X}$, while the other player $\boldsymbol{y}$ intends to maximize $f(\boldsymbol{x}, \boldsymbol{y})$ inside $\mathcal{Y}$. Applications of this problem include generative adversarial networks (GANs) [61], hierarchical reinforcement learning [33], adversarial learning [150], proximal gradient TD learning [106], fair statistical inference [51, 107], synthetic gradients [70], imaginary agents [162] and many more. With the wide applications of this problem, it is important to develop efficient algorithms that probably find the optimal points of Eqn. (1.1). Gradient-based methods, especially gradient descent-ascent (GDA), are widely used in practice to solve these problems. GDA alternates between a gradient ascent steps on $\boldsymbol{x}$ and a gradient descent steps on $\boldsymbol{y}$. We continue with an overview of min-max optimization in different settings.

1.1.1 Strongly-Convex Primal-Dual Formulation

The primal-dual convex-concave saddle point problem is of the form:

$$\min_{\boldsymbol{x}\in\mathcal{X}\subseteq\mathbb{R}^d}\max_{\boldsymbol{y}\in\mathcal{Y}\subseteq\mathbb{R}^n} g(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-f(\boldsymbol{y}). \tag{1.2}$$

This setting is directly motivated by a wide range of applications including regularized and constrained empirical risk minimization [170], a class of unsupervised learning [168], policy evaluation [46], robust optimization [19], and so forth.

With the primal-dual formulation, prior work focuses on unconstrained problems. For the case when f and g are both strongly convex, it has been understood for long that primal-dual gradient-type methods converge linearly [25]. Further, [47, 161] showed that GDA achieves a linear convergence rate when g is convex and f is strongly convex.

Under the separable assumption that $f=\frac{1}{n}\sum_i f_i$, [178] introduce a novel stochastic primal-dual coordinate method (SPDC), which with acceleration achieves a time complexity of $\mathcal{O}(nd\,(1+\sqrt{\kappa/n})\log(1/\epsilon))$[1], matching that of accelerated stochastic dual coordinate descent methods.

1.1.2 Convex Min-Max Games

Arguably, one of the most celebrated theorems and a founding stone in Game Theory is the minimax theorem by Von Neumann [160]. It states that

$$\min_{\boldsymbol{x}\in\Delta_n}\max_{\boldsymbol{y}\in\Delta_m} f(\boldsymbol{x},\boldsymbol{y})=\max_{\boldsymbol{y}\in\Delta_m}\min_{\boldsymbol{x}\in\Delta_n} f(\boldsymbol{x},\boldsymbol{y}), \tag{1.3}$$

[1]Here κ is the condition number of the primal form $g(\boldsymbol{x})+\frac{1}{n}\sum_i f_i^*(\boldsymbol{a}_i^\top\boldsymbol{x})$

where $f : \Delta_n \times \Delta_m \to \mathbb{R}$ is convex in $\boldsymbol{x}$, concave in $\boldsymbol{y}$. The aforementioned result holds for any convex compact sets $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}^m$. The min-max theorem reassures us that an equilibrium always exists in bilinear games (1.2) or its convex-concave analog ($f(\boldsymbol{x}, \boldsymbol{y})$ is interpreted as the payment of the $\boldsymbol{x}$ player to the $\boldsymbol{y}$ player). Equilibrium is a pair of randomized strategies $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ such that neither player can improve their payoff by unilaterally changing their distribution.

Soon after the appearance of the minimax theorem, research was focused on dynamics for solving min-max optimization problems by having the min and max players of (3.1) run a simple online learning procedure. In the online learning framework, at time t, each player chooses a probability distribution ($\boldsymbol{x}^t, \boldsymbol{y}^t$ respectively) simultaneously depending *only* on the past choices of both players (i.e., $\boldsymbol{x}^1, ..., \boldsymbol{x}^{t-1}, \boldsymbol{y}^1, ..., \boldsymbol{y}^{t-1}$) and experiences payoff that depends on choices $\boldsymbol{x}^t, \boldsymbol{y}^t$.

An early method, proposed by Brown [18] and analyzed by Robinson [142], was fictitious play. Later on, researchers discover several learning robust algorithms converging to minimax equilibrium at faster rates, see [21]. This class of learning algorithms, are the so-called "no-regret" and include the Multiplicative Weights Update method [11] and Follow the regularized leader.

Average Iterate Convergence: Despite the rich literature on no-regret learning, most of the known results have the feature that min-max equilibrium is shown to be attained only by the time *average*. This means that the trajectory of a no-regret learning method $(\boldsymbol{x}^t, \boldsymbol{y}^t)$ has the property that $\frac{1}{t}\sum_{\tau \le t} (\boldsymbol{x}^\tau)^\top A \boldsymbol{y}^\tau$ converges to the equilibrium of (3.1), as $t \to \infty$. Unfortunately, that does not mean that the last iterate $(\boldsymbol{x}^t, \boldsymbol{y}^t)$ converges to an equilibrium, it commonly diverges or cycles. One such example is the well-known Multiplicative Weights Update Algorithm, the time average of which is known to converge to an equilibrium, but the actual trajectory cycles towards the boundary of the simplex ([14]). This is even true for the vanilla Gradient Descent/Ascent, where one can

show for even bilinear landscapes (unconstrained case) the last iterate fails to converge [39].

Motivated by the training of Generative Adversarial Networks (GANs), the last couple of years researchers have focused on designing and analyzing procedures that exhibit *last iterate* convergence (or pointwise convergence) for zero-sum games. This is crucial for training GANs, the landscapes of which are typically non-convex non-concave and averaging now as before do not give many guarantees (e.g., note that Jensen's inequality is not applicable anymore). In [39, 103] the authors show that a variant of Gradient Descent/Ascent, called Optimistic Gradient Descent/Ascent has last iterate convergence for the case of bilinear functions $\boldsymbol{x}^\top A\boldsymbol{y}$ where $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$ (this is called the unconstrained case, since there are no restrictions on the vectors). Later on, [40] generalized the above result with simplex constraints, where the online method that the authors analyzed was Optimistic Multiplicative Weights Update. In [113], it is shown that Mirror Descent with extra gradient computation converges pointwise for a class of zero-sum games that includes the convex-concave setting (with arbitrary constraints), though their algorithm does not fit in the online no-regret framework since it uses information twice about the payoffs before it iterates. Last but not least there have appeared other works that show pointwise convergence for other settings (see [131, 41] and [1] and references therein) to stationary points (but not local equilibrium solutions).

1.1.3 Non-Convex Games

Non-convex and non-concave problems are the most general and indisputably the hardest setting. With this general form, finding its equilibria is at least as hard as standard minimization problems. This problem is therefore "hopelessly impractical to solve" in general [146], because it is PPAD hard [42]. To tackle this problem, a starting point is to formally define the local min-max points such that algorithms with local updates (i.e. GDA and many other commonly used

first/second-order algorithms) could possibly find them. Jin et al. take an initial step in the machine learning community to formalize the local min-max point in [72]. However, local min-max points are not guaranteed to exist and therefore in general GDA will not always be effective to find them.

Stronger Conditions: Fortunately, with stricter conditions, one may still possibly derive convergence guarantees, either locally or globally. Under some strong conditions, it is established in [29] that GDA dynamics converges locally to Nash equilibria. While the work in [41] study min-max optimization (or zero-sum games), a much more general setting of nonzero-sum games and multi-player games is considered in [109]. They have established that stable limit points of GDA are not necessarily Nash equilibria. However, second-order methods have proven useful in the sense of their stable fixed points are exactly Nash equilibria, as shown in [4, 109]. Under nonconvex but concave setting, an algorithm combining approximate maximization over $\boldsymbol{y}$ and a proximal gradient method for $\boldsymbol{x}$ is proposed in [137] to show convergence to stationary points.

Specific Applications: Meanwhile, for some specific problems, one could still possibly prove global convergence, or GDA heuristics have been proved effective in practice due to the good problem structure. Specifically, some strong assumptions have been investigated in the setting of GAN training [67, 121] to ensure that Nash equilibria are stable fixed points of GDA. When the objective satisfies a variational inequality, by solving some strong variational inequality problems, [104] proposes a proximal algorithm with convergence to stationary points. We show that stochastic GDA learns the optimal distribution with one-layer generators using Wasserstein-GANs [93].

For adversarial training (AT), the update on the first player (i.e., the attacker) is constrained to move in some small ball (i.e., the threat model). AT requires to achieve maximum value in the

inner loop and the problem will become a simple minimization problem. In this case, AT heuristics with PGD attacks are effective in practice. We also propose some heuristic for the problems where [28]. Some recent work [55] studies the dynamics under the NTK (Neural Tangent Kernel) regime.

1.2 Organization and Contributions

This thesis focuses on four concrete settings in min-max optimization. We gradually go from the simplest setting, i.e., (strongly) convex-concave to a more general setting where it becomes unclear whether a simple first-order algorithm will find the optimal min-max point.

For strongly-convex and strongly concave setting, GDA and its variants have proven effective. In Chapter 2, we establish the situations where reformulating a single minimization problem into a two-player game improves the convergence speed to reach equilibrium [97, 100]. The min-max formulation enables us to exploit the underlying problem structure such as sparsity or low rank, and the cost of our method depends only on the structural complexity of the solutions instead of the ambient dimension.

Despite the popularity of the GDA algorithm, it fails to converge even for simple bilinear zero-sum games [39]. But for a convex-concave setting, this issue could be fixed by some small adjustments like extra-gradient [118] or adding negative momentum [39]. However, the problem is generally much harder for constrained problems [40]. On the other hand, for real games we care about mixed strategies rather than single actions. For mixed strategies we generally represent θ and ω as probability density over possible actions. Therefore it is more important to study constrained problems, especially with simplex constraints that represent categorical distributions.

In Chapter 3, we proposed the **optimistic multiplicative weight update** algorithm that

provably exhibits local convergence to equilibrium points, for convex-concave min-max games with simplex constraints [94]. It is established on a careful analysis of the dynamical system induced by our algorithm.

In Chapter 4 we studied the **training dynamics of generative adversarial networks** (GANs), which is a non-convex/non-concave game. Specifically, we show that with stochastic gradient descent, we can learn an optimal generator for one-layer generative networks with polynomial time and sample complexity [93].

This thesis is based on my existing work [97, 100, 94, 93]. I have also studied other topics during my PhD, including matrix analysis [158, 99, 174], distributed learning [155, 179], neural network architecture [175], adversarial attack/robustness [96, 169, 28], data mining [98, 165, 95, 172, 171], compressed sensing [92, 164], and representation learning [48].

Chapter 2

(Constrained) Strongly Convex-Concave Objective: On Exploiting the Structural Complexity

We consider the convex-concave saddle point problem $\min_{x\in C}\max_{y\in\mathcal{Y}} g(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-f(\boldsymbol{y})$ where the constraint C or the regularizer g enforce some underlying structure on the optimal variable $\boldsymbol{x}^*$ such as sparsity or low rank. We propose a class of algorithms that fully exploit the problem structure and reduce the per-iteration cost while maintaining linear convergence rate. The per iteration cost of our methods depend on the structural complexity of the solution (i.e. sparsity/low-rank) instead of the ambient dimension. We empirically show that our algorithm outperforms the state-of-the-art methods on (multi-class) classification tasks.[1]

2.1 Introduction

We consider optimization problems of the form:

$$\min_{x\in C\subseteq\mathcal{X}}\max_{y\in\mathcal{Y}}:\ \ g(\boldsymbol{x})+\boldsymbol{y}^\top A\boldsymbol{x}-f(\boldsymbol{y}),$$

motivated by a wide range of applications including regularized and constrained empirical risk minimization [170], a class of unsupervised learning [168], policy evaluation [46], robust optimization [19] and so forth. Particularly, we are interested in problems whose solution has special "simple" structure like low-rank or sparsity. The sparsity constraint applies to large-scale multiclass/multi-label classification, low-degree polynomial data mapping [23], random feature kernel machines [138], and Elastic Net [180]. Motivated by recent applications in low-rank multi-class SVM, phase retrieval, matrix completion, affine rank minimization and other problems (e.g.,

[1]This work is based on the following published conference papers:
1. Qi Lei, Jiacheng Zhuo, Constantine Caramanis, Inderjit S Dhillon, Alexandros G Dimakis. "Primal-Dual Block Frank-Wolfe", Proc. of Neural Information Processing Systems (NeurIPS) 2019 [100]
2. Qi Lei, Enxu Yan, Chao-yuan Wu, Pradeep Ravikumar, Inderjit Dhillon, "Doubly Greedy Primal-Dual Coordinate Methods for Sparse Empirical Risk Minimization", Proc. of International Conference of Machine Learning (ICML), 2017 [97]

[49, 136, 6, 20, 179]), we also consider settings where the constraint $\boldsymbol{x} \in C$ (e.g., trace norm ball) while convex, may be difficult to project onto. A wish-list for this class of problems would include an algorithm that (1) exploits the simple structure of the solution, (2) achieves linear convergence for smooth and strongly convex problems, (3) does not pay a heavy price for the projection step.

For constrained case where C is nuclear norm or ℓ_1 norm bound, we propose a Frank-Wolfe type algorithm. For unconstrained problem where $C = \mathcal{X}$ and g contains ℓ_1 regularizer, we propose a doubly greedy update routine. Our proposals attain these three goals simultaneously. However, this does not come without challenge:

Notice the considered saddle-point problem is equivalent to a simple minimization on $g(\boldsymbol{x}) + f^*(A\boldsymbol{x})$ (normally referred as the primal form) where f^* is the convex conjugate of f. However, prior studies that directly optimize on the primal form do not benefit from the simple structures of the optimal solution. We argue that the saddle point formulation accordingly enables us to achieve the first goal. Specifically, we show that $\boldsymbol{y}$ guides the search of the most important sparse or low rank directions to update in $\boldsymbol{x}$, and vice versa for $\boldsymbol{x}$. Such structural updates cost much fewer computations but achieve comparable progress as full updates.

On the other hand, for problems like phase retrieval and ERM for multi-label multi-class classification, the gradient computation requires large matrix multiplications. This dominates the per-iteration cost, and the existing FW type methods do not asymptotically reduce time complexity per iteration, even without paying the expensive projection step. Meanwhile, for simpler constraints like the ℓ_1 norm ball or the simplex, it is unclear if FW can offer any benefits compared to other methods. Moreover, as is generally known, FW suffers from sub-linear convergence rate even for well-conditioned problems that enjoy strong convexity and smoothness.

2.2 Related Work

Frank-Wolfe Type Methods We review relevant algorithms that improve the overall performance of Frank-Wolfe type methods. Such improvements are roughly obtained for two reasons: the enhancement on convergence speed and the reduction on iteration cost. Very few prior works benefit in both.

Nesterov's acceleration has proven effective as in Stochastic Condition Gradient Sliding (SCGS) [87] and other variants [163, 117, 56]. Restarting techniques dynamically adapt to the function geometric properties and fills in the gap between sublinear and linear convergence for FW method [80]. Some variance reduced algorithms obtain linear convergence as in [66], however, the number of inner loops grows significantly and hence the method is not computationally efficient.

Linear convergence has been obtained specifically for polytope constraints like [122], as well as the work proposed in [85, 60] that use the Away-step Frank Wolfe and Pair-wise Frank Wolfe, and their stochastic variants. One recent work [5] focuses on trace norm constraints and proposes a FW-type algorithm that yields similar progress as projected gradient descent per iteration but is almost projection free. However, in many applications where gradient computation dominates the iteration complexity, the reduction on projection step doesn't necessarily produce asymptotically better iteration costs.

The sparse update introduced by FW steps was also appreciated by [86], where they conducted dual updates with a focus on SVM with polytope constraint. Their algorithm yields low iteration costs but still suffer from sub-linear convergence.

Primal-Dual Formulation With the primal-dual formulation, prior work focuses on unconstrained problems. For the case when f and g are both strongly convex, it has been understood for

long that primal-dual gradient-type methods converge linearly [25]. Further, [47, 161] showed that GDA achieves a linear convergence rate when g is convex and f is strongly convex.

Under the separable assumption that $f = \frac{1}{n}\sum_i f_i$, [178] introduce a novel stochastic primal-dual coordinate method (SPDC), which with acceleration achieves a time complexity of $\mathcal{O}(nd\,(1+\sqrt{\kappa/n})\log(1/\epsilon))$[2], matching that of accelerated stochastic dual coordinate descent methods.

However, in practice, SPDC could lead to more expensive computations for sparse data matrices due to dense updates. For some special choices of the model, [178] provided efficient implementation for sparse feature structures, but the average update time for each coordinate is still much longer than that of dual coordinate descent. Moreover, they cannot exploit intermediate sparse iterates by methods such as shrinking technique [68].

We note that the mentioned algorithms only show worse than or simply match the overall complexity compared to conventional methods that optimize on the primal form directly. Therefore we raise the following question: *Does the primal-dual formulation have other good properties that could be leveraged to improve optimization performance?*

For instance, some recent work with the primal-dual formulation updates stochastically sampled coordinates [173], which has a reduced cost per iteration, provided the data admits a low-rank factorization or when the proximal mapping for primal and dual variables are relatively computational expensive, which however may not hold in practice, so that the the noise caused by this preprocessing could hurt test performance. Moreover, even when their assumptions hold, their low-rank matrix factorization step itself may dominate the total computation time.

[2]Here κ is the condition number of the primal form $g(\boldsymbol{x}) + \frac{1}{n}\sum_i f_i^*(\boldsymbol{a}_i^\top \boldsymbol{x})$

Our contributions. In this work we tackle the challenges by exploiting the special structure induced by the constraints and FW steps. We propose a generalized variant of FW that we call Primal-Dual Block Generalized Frank Wolfe. The main advantage is that the computational complexity depends only on the sparsity of the solution, rather than the ambient dimension, i.e. it is *dimension free*. This is achieved by conducting *partial updates* in each iteration, i.e., sparse updates for ℓ_1 and low-rank updates for the trace norm ball. While the benefits of *partial updates* is unclear for the original problem, we show in this work how they significantly benefit a primal-dual reformulation. This reduces the per iteration cost to roughly a ratio of $\frac{s}{d}$ compared to naive Frank-Wolfe, where s is the sparsity (or rank) of the optimal solution, and d is the feature dimension. Meanwhile, the per iteration progress of our proposal is comparable to a full gradient descent step, thus retaining linear convergence rate.

For strongly convex and smooth f and g we show that our algorithm achieves linear convergence with per-iteration cost sn over ℓ_1-norm ball, where s upper bounds the sparsity of the primal optimal. Specifically, for sparse ERM with smooth hinge loss or quadratic loss with ℓ_2 regularizer, our algorithm yields an overall $\mathcal{O}(s(n+\kappa)\log\frac{1}{\epsilon})$ time complexity to reach ϵ duality gap, where κ is the condition number (smoothness divided by strong convexity). Our theory has minimal requirements on the data matrix A.

Experimentally we observe our method yields significantly better performance compared to prior work, especially when the data dimension is large and the solution is sparse. Therefore we achieve the state-of-the-art performance both in time complexity and in practice measured by CPU time, for regularized ERM with smooth hinge loss and matrix sensing problems.

2.3 Preliminary

Notation. We briefly introduce the notation used throughout the paper. We use bold lower case letter to denote vectors, capital letter to represent matrices. $\|\cdot\|$ is ℓ_2 norm for vectors and Frobenius norm for matrices unless specified otherwise. $\|\cdot\|_*$ indicates the trace norm for a matrix.

We say a function f is α strongly convex if $f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \langle \boldsymbol{g}, \boldsymbol{y} - \boldsymbol{x} \rangle + \frac{\alpha}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2$, where $\boldsymbol{g} \in \partial f(\boldsymbol{x})$ is any sub-gradient of f. Similarly, f is β-smooth when $f(\boldsymbol{y}) \leq f(\boldsymbol{x}) + \langle \boldsymbol{g}, \boldsymbol{y} - \boldsymbol{x} \rangle + \frac{\beta}{2}\|\boldsymbol{y} - \boldsymbol{x}\|^2$. We use f^* to denote the convex conjugate of f, i.e., $f^*(\boldsymbol{y}) \triangleq \max_{\boldsymbol{x}} \langle \boldsymbol{x}, \boldsymbol{y} \rangle - f(\boldsymbol{x})$. Some more parameters are problem-specific and are defined when needed.

Primal-Dual Formulation. Note that the problem we are tackling is as follows:

$$\min_{\boldsymbol{x} \in C} \{P(\boldsymbol{x}) \equiv f^*(A\boldsymbol{x}) + g(\boldsymbol{x})\}, \tag{2.1}$$

We first focus on the setting where $\boldsymbol{x} \in \mathbb{R}^d$ is a vector and C is the ℓ_1-norm ball. This form covers general classification or regression tasks with f being some loss function and g being a regularizer. Extension to matrix optimization over a trace norm ball is introduced in Section 2.5.1.

Even with the constraint, we could reform (2.1) as a primal-dual convex-concave saddle point problem:

$$(2.1) \quad \Leftrightarrow \quad \min_{\boldsymbol{x} \in C} \max_{\boldsymbol{y} \in \mathcal{Y}} \{\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) \equiv g(\boldsymbol{x}) + \langle \boldsymbol{y}, A\boldsymbol{x} \rangle - f(\boldsymbol{y})\}, \tag{2.2}$$

or its dual formulation:

$$(2.1) \Leftrightarrow \max_{\boldsymbol{y}} \left\{ D(\boldsymbol{y}) := \min_{\boldsymbol{x} \in C} \{g(\boldsymbol{x}) + \langle \boldsymbol{y}, A\boldsymbol{x} \rangle\} - f(\boldsymbol{y}) \right\}. \tag{2.3}$$

Notice (2.3) is not guaranteed to have an explicit form. Therefore some existing FW variants like [86] that optimizes over (2.3) may not apply. Instead, we directly solve the convex concave problem

(2.2) and could therefore solve more general problems, including complicated constraint like trace norm.

Since the computational cost of the gradient $\nabla_{\boldsymbol{x}}\mathcal{L}$ and $\nabla_{\boldsymbol{y}}\mathcal{L}$ is dominated by computing $A^\top \boldsymbol{y}$ and $A\boldsymbol{x}$ respectively, *sparse updates* could reduce computational costs by a ratio of roughly $\mathcal{O}(d/s)$ for updating $\boldsymbol{x}$ and $\boldsymbol{y}$ while achieving good progress.

2.3.1 A Theoretical Vignette

In this section, we review the previous methods that achieve linear convergence while conducting only partial (low rank/sparse) updates on the learned variables.

To elaborate the techniques we use to obtain the linear convergence for our Frank-Wolfe type algorithm, we consider the ℓ_1 norm constrained problem as an illustrating example:

$$\arg\min_{\boldsymbol{x}\in\mathbb{R}^d, \|\boldsymbol{x}\|_1\leq\tau} f(\boldsymbol{x}), \tag{2.4}$$

where f is L-smooth and μ-strongly convex. If we invoke the Frank Wolfe algorithm, we compute

$$\boldsymbol{x}^{(t)} \leftarrow (1-\eta)\boldsymbol{x}^{(t-1)} + \eta\tilde{\boldsymbol{x}}, \qquad \text{where } \tilde{\boldsymbol{x}} \leftarrow \arg\min_{\|\boldsymbol{x}\|_1\leq\tau} \langle \nabla f(\boldsymbol{x}^{(t-1)}), \boldsymbol{x}\rangle. \tag{2.5}$$

Even when the function f is smooth and strongly convex, (2.5) converges sublinearly. As inspired by [5], if we assume the optimal solution is s-sparse, we can enforce a sparse update while maintaining linear convergence by a mild modification on (2.5):

$$\boldsymbol{x}^{(t)} \leftarrow (1-\eta)\boldsymbol{x}^{(t-1)} + \eta\tilde{\boldsymbol{x}}, \text{ where } \tilde{\boldsymbol{x}} \leftarrow \arg\min_{\|\boldsymbol{x}\|_1\leq\tau, \|\boldsymbol{x}\|_0\leq s} \{\langle \nabla f(\boldsymbol{x}^{(t-1)}), \boldsymbol{x}\rangle + \frac{L}{2}\eta\|\boldsymbol{x}^{(t-1)} - \boldsymbol{x}\|_2^2\}. \tag{2.6}$$

We also call this new practice block Frank-Wolfe as in [5]. The proof of convergence can be completed within three lines. Let $h_t = f(\boldsymbol{x}^{(t)}) - \min_{\boldsymbol{x}} f(\boldsymbol{x})$.

$$h_t = f(\boldsymbol{x}^{(t-1)} + \eta(\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(t-1)})) - \min_{\boldsymbol{x}} f(\boldsymbol{x})$$

$$
\begin{aligned}
&\leq h_{t-1} + \eta\langle \nabla f(\boldsymbol{x}^{(t-1)}), \tilde{\boldsymbol{x}} - \boldsymbol{x}^{(t-1)}\rangle + \frac{L}{2}\eta^2\|\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(t-1)}\|^2 && \text{(Smoothness of } f\text{)} \\
&\leq h_{t-1} + \eta\langle \nabla f(\boldsymbol{x}^{(t-1)}), \boldsymbol{x}^* - \boldsymbol{x}^{(t-1)}\rangle + \frac{L}{2}\eta^2\|\boldsymbol{x}^* - \boldsymbol{x}^{(t-1)}\|^2 && \text{(Definition of } \tilde{\boldsymbol{x}}\text{)} \\
&\leq (1 - \eta + \frac{L}{\mu}\eta^2)h_{t-1} && \text{(by convexity and } \mu\text{-strong convexity of } f\text{)} \qquad (2.7)
\end{aligned}
$$

Therefore, when $\eta = \frac{\mu}{2L}$, $h_{t+1} \leq (1 - \frac{\mu}{4L})^t h_1$ and the iteration complexity is $\mathcal{O}(\frac{L}{\mu}\log(1/\epsilon))$ to achieve ϵ error.

Similarly, with greedy coordinate descent algorithm, we simply remove the additional constraint and conduct the following update:

$$
\boldsymbol{x}^{(t)} \leftarrow (1-\eta)\boldsymbol{x}^{(t-1)} + \eta\tilde{\boldsymbol{x}}, \text{ where } \tilde{\boldsymbol{x}} \leftarrow \underset{\|\boldsymbol{x}\|_0 \leq s}{\arg\min}\{\langle \nabla f(\boldsymbol{x}^{(t-1)}), \boldsymbol{x}\rangle + \frac{L}{2}\eta\|\boldsymbol{x}^{(t-1)} - \boldsymbol{x}\|_2^2\}. \quad (2.8)
$$

With exact the same analysis, we note that GCD also achieves linear convergence with sufficiently large s.

For both methods, we note that to search for the sparse update, one requires to compute the full gradient. This costs the same computational complexity as (Projected) Gradient Descend, without further assumption of f. Luckily, with the sparse updates, it is possible to improve the iteration complexity, while maintaining the linear convergence rate. In order to differentiate, we name the sparse update nature of (2.6) as *partial update*.

Next we elaborate the situations when one benefits from *partial updates*. Consider a quadratic function: $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top A\boldsymbol{x}$, whose gradient is $A\boldsymbol{x}$ for symmetric A. As $\tilde{\boldsymbol{x}}$ is sparse, One can maintain the value of the gradient efficiently [99]: $A\boldsymbol{x}^{(t)} \equiv (1-\eta)A\boldsymbol{x}^{(t-1)} + \eta A_{I,:}\tilde{\boldsymbol{x}}$, where I is the support set of $\tilde{\boldsymbol{x}}$. We therefore reduce the complexity of one iteration to $\mathcal{O}(sd)$, compared to $\mathcal{O}(d^2)$ with PGD. Similar benefits hold when we replace $\boldsymbol{x}$ by a matrix X and conduct a low-rank update on X. The benefit of *partial update* is not limited to quadratic functions. Next we show that

for a class of composite function, we are able to take the full advantage of the *partial update*, by taking a primal-dual re-formulation.

2.4 Methodology

With the primal-dual formulation, we are ready to introduce our algorithm. The idea is simple: for primal variable $\boldsymbol{x}$, we conduct block Frank-Wolfe or greedy coordinate descent respectively for constrained and unconstrained cases. Meanwhile, for the dual variable $\boldsymbol{y}$ we conduct greedy coordinate ascent method to select and update k coordinates (k determined later). We selected coordinates that allow the largest step, which is usually referred as a Gauss-Southwell rule denoted by **GS-r** [129]. We have the following assumptions on f and g:

Assumption 2.4.1. *We assume the functions satisfy the following properties:*

- *Each loss function f is convex and β-smooth, and is α strongly convex over some convex set (could be $\mathbb{R}$), and linear otherwise.*
- $\|\boldsymbol{a}_i\|_2 \leq R, \forall i.$
- *g is μ-strongly convex and L-smooth.*

Suitable loss functions f include smooth hinge loss [145] and quadratic loss function. Relevant applications covered are Support Vector Machine (SVM) with smooth hinge loss, elastic net [180], matrix sensing, linear regression problem with quadratic loss and so forth.

Algorithm 1 Primal-Dual Block Generalized Frank-Wolfe Method for ℓ_1 Norm Ball

Input: Training data $A \in \mathbb{R}^{n \times d}$, primal and dual step size $\eta, \delta > 0$.
Initialize: $\boldsymbol{x}^{(0)} \leftarrow 0 \in \mathbb{R}^d$, $\boldsymbol{y}^{(0)} \leftarrow 0 \in \mathbb{R}^n$, $\boldsymbol{w}^{(0)} \equiv A\boldsymbol{x} = 0 \in \mathbb{R}^n$, $\boldsymbol{z}^{(0)} \equiv A^\top \boldsymbol{y} = 0 \in \mathbb{R}^d$
for $t = 1, 2, \cdots, T$ **do**
    Use Block Frank Wolfe to update the primal variable:

$$\tilde{\boldsymbol{x}} \leftarrow \underset{\|\boldsymbol{x}\|_1 \le \lambda, \|\boldsymbol{x}\|_0 \le s}{\arg\min} \{\langle \boldsymbol{z}^{(t-1)} + \nabla g(\boldsymbol{x}^{(t-1)}), \boldsymbol{x} \rangle + \frac{L}{2}\eta \|\boldsymbol{x} - \boldsymbol{x}^{(t-1)}\|^2\} \tag{2.9}$$

$$\boldsymbol{x}^{(t)} \leftarrow (1-\eta)\boldsymbol{x}^{(t-1)} + \eta \tilde{\boldsymbol{x}}$$

    Update w to maintain the value of $A\boldsymbol{x}$:

$$\boldsymbol{w}^{(t)} \leftarrow (1-\eta)\boldsymbol{w}^{(t-1)} + \eta A \Delta \boldsymbol{x} \tag{2.10}$$

    Consider the potential dual update:

$$\tilde{\boldsymbol{y}} = \underset{\boldsymbol{y}'}{\arg\max} \left\{ \langle \boldsymbol{w}^{(t)}, \boldsymbol{y}' \rangle - f(\boldsymbol{y}') - \frac{1}{2\delta} \|\boldsymbol{y}' - \boldsymbol{y}^{(t-1)}\|^2 \right\}. \tag{2.11}$$

    Choose greedily the dual coordinates to update: let $I^{(t)}$ be the top k coordinates that maximize

$$|\tilde{y}_i - y_i^{(t-1)}|, i \in [n].$$

    Update the dual variable accordingly:

$$y_i^{(t)} \leftarrow \begin{cases} \tilde{y}_i & \text{if } i \in I^{(t)} \\ y_i^{(t-1)} & \text{otherwise.} \end{cases} \tag{2.12}$$

    Update z to maintain the value of $A^\top \boldsymbol{y}$

$$\boldsymbol{z}^{(t)} \leftarrow \boldsymbol{z}^{(t-1)} + A_{:,I^{(t)}}^\top (\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}) \tag{2.13}$$

end for
Output: $\boldsymbol{x}^{(T)}, \boldsymbol{y}^{(T)}$

2.4.1 Primal-Dual Block Generalized Frank-Wolfe

The formal algorithm is presented in Algorithm 1. As $\mathcal{L}(\boldsymbol{x}, \boldsymbol{y})$ is μ-strongly convex and L-smooth with respect to $\boldsymbol{x}$, we set the primal learning rate $\eta = \frac{\mu}{2L}$ according to Section 2.3.1.

Meanwhile, the dual learning rate δ is set to balance its effect on the dual progress as well as the primal progress. We specify it in the theoretical analysis part.

The computational complexity for each iteration in Algorithm 1 is $\mathcal{O}(ns)$. Both primal and dual update could be viewed as roughly three steps: coordinate selection, variable update, and maintaining $A^T\boldsymbol{y}$ or $A\boldsymbol{x}$. The coordinate selection as Eqn. (2.9) for primal and the choice of $I^{(t)}$ for dual variable respectively take $\mathcal{O}(d)$ and $\mathcal{O}(n)$ on average if implemented with the quick selection algorithm. The variable update costs $\mathcal{O}(d)$ and $\mathcal{O}(n)$. The dominating cost is to maintain $A\boldsymbol{x}$ as in Eqn. (2.10) that takes $\mathcal{O}(ns)$, and $\mathcal{O}(dk)$ of maintaining $A^\top\boldsymbol{y}$ as in Eqn. (2.13). To balance the time budget for primal and dual step, we set $k = ns/d$ and achieve an overall complexity of $\mathcal{O}(ns)$ per iteration.

For unconstrained problems, we simply replace the Eqn. (2.9) in Step 4 with the unconstrained version (2.8).

2.5 Theoretical Analysis

We derive convergence analysis under Assumption 2.4.1. The derivation consists of the analysis on the primal progress, the balance of the dual progress, and their overall effect.

Define the primal gap as $\Delta_p^{(t)} \triangleq \mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})$, where $\bar{\boldsymbol{x}}^{(t)}$ is the primal optimal solution such that the dual $D(\boldsymbol{y}^{(t)}) = \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})$, and is sparse enforced by the ℓ_1 constraint. The dual gap is $\Delta_d^{(t)} \triangleq D^* - D(\boldsymbol{y}^{(t)})$. We analyze the convergence rate of duality gap $\Delta^{(t)} \equiv \max\{1, (\beta/\alpha - 1)\}\Delta_p^{(t)} + \Delta_d^{(t)}$.

Primal progress: Firstly, similar to the analysis in Section 2.3.1, we could derive that

primal update introduces a sufficient descent as in Lemma A.1.2.

$$\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) \leq -\frac{\eta}{2}\Delta_p^{(t)}.$$

Dual progress: With the **GS-r** rule to carefully select and update the most important k coordinates in the dual variable in (2.11), we are able to derive the following result on dual progress that diminishes dual gap as well as inducing error.

$$-\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \leq -\frac{k\delta}{n\beta}\Delta_d^{(t)} + \frac{k\delta}{n^2}R\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2$$

Refer to Lemma A.1.5 for details.

Primal Dual progress: The overall progress evolves as:

$$\Delta^{(t)} - \Delta^{(t-1)} \leq \overbrace{\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})}^{\text{primal progress}} - \frac{1}{4\delta}\overbrace{\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2}^{\text{dual progress}} + \frac{3\delta Rk}{2n^2}\overbrace{\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2}^{\text{primal hindrance}}.$$

In this way, we are able to connect the progress on duality gap with constant fraction of its value, and achieve linear convergence:

Theorem 2.5.1. *Given a function $P(\boldsymbol{x}) = f^*(A\boldsymbol{x}) + g(\boldsymbol{x})$ that satisfies Assumption 2.4.1. Set s to upper bound the sparsity of the primal optimal $\bar{\boldsymbol{x}}^{(t)}$, and learning rates $\eta = \frac{\mu}{2L}, \delta = \frac{1}{k}(\frac{L}{\mu\beta} + \frac{5\beta R}{2\alpha\mu}(1 + 4\frac{L}{\mu}))^{-1}$. The duality gap $\Delta^{(t)} = \max\{1, \frac{\beta}{\alpha} - 1\}\Delta_p^{(t)} + \Delta_d^{(t)}$ generated by Algorithm 1 takes $\mathcal{O}(\frac{L}{\mu}(1 + \frac{\beta}{\alpha}\frac{R\beta}{\mu})\log\frac{1}{\epsilon})$ iterations to achieve ϵ error. The overall complexity is $\mathcal{O}(ns\frac{L}{\mu}(1 + \frac{\beta}{\alpha}\frac{R\beta}{\mu})\log\frac{1}{\epsilon})$.*

For our target applications like elastic net, or ERM with smooth hinge loss, the loss function is separable: $f(\boldsymbol{y}) = \frac{1}{n}\sum_i f_i(\boldsymbol{y})$. In this case, the primal-dual form for $f^*(A\boldsymbol{x})$ becomes $\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = g(\boldsymbol{x}) + \frac{1}{n}\boldsymbol{y}^\top A\boldsymbol{x} - \frac{1}{n}\sum_i f_i(y_i)$, we are able to connect the time complexity to the condition number of the primal form:

Corollary 2.5.2. *Given an objective* $P(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i^*(\boldsymbol{a}_i^\top \boldsymbol{x}) + g(\boldsymbol{x})$*, with a smooth hinge loss or quadratic loss* f_i^* *that is* β*-smooth, and* ℓ_2 *regularizer* $g = \frac{\mu}{2}\|\boldsymbol{x}\|^2$*. Define the condition number* $\kappa = \frac{\beta R}{\mu}$*. Setting* s *upper bounds the sparsity of the primal optimal* $\bar{\boldsymbol{x}}^{(t)}$*, and learning rates* $\eta = \frac{1}{2}, \delta = \frac{1}{k}(\frac{1}{n\beta} + \frac{25R}{2\mu n^2})^{-1}$*, the duality gap* $\Delta^{(t)}$ *takes* $\mathcal{O}((1+\frac{\kappa}{n})\log\frac{1}{\epsilon})$ *iterations to achieve* ϵ *error. The overall complexity is* $\mathcal{O}(s(n+\kappa)\log\frac{1}{\epsilon})$.

Remark 2.5.1. *Both Theorem 2.5.1 and Corollary 2.5.2 cover the unconstrained when we replace block Frank-Wolfe with Greedy Cooordinate Descent steps trivially.*

Our derivation of overall complexity implicitly requires $ns \geq d$ by setting $k = sd/n \geq 1$. This is true for our considered applications like SVM. Otherwise we choose $k = 1$ and the complexity becomes $\mathcal{O}(\max\{d, ns\}(1+\frac{\kappa}{n})\log\frac{1}{\epsilon})$.

In Table 2.1, we briefly compare the time complexity of our algorithm with some benchmark algorithms: (1) Accelerated Projected Gradient Descent (PGD) (2) Frank-Wolfe algorithm (FW) (3) Stochastic Variance Reduced Gradient (SVRG) [74] (4) Stochastic Conditional Gradient Sliding (SCGS) [87] and (5) Stochastic Variance-Reduced Conditional Gradient Sliding (STORC) [66]. The comparison is not thorough but intends to select constrained optimization that improves the overall complexity from different perspective. Among them, accelerated PGD improves conditioning of the problem, while SCGS and STORC reduces the dependence on number of samples. In the experimental session we show that our proposal outperforms the listed algorithms under various conditions.

Algorithm	Per Iteration Cost	Iteration Complexity
Frank Wolfe	$\mathcal{O}(nd)$	$\mathcal{O}(\frac{1}{\epsilon})$
Accelerated PGD [127]	$\mathcal{O}(nd)$	$\mathcal{O}(\sqrt{\kappa}\log\frac{1}{\epsilon})$
SVRG [74]	$\mathcal{O}(nd)$	$\mathcal{O}((1+\kappa/n)\log\frac{1}{\epsilon})$
SCGS [87]	$\mathcal{O}(\kappa^2\frac{\#\text{iter}^3}{\epsilon^2}d)$	$\mathcal{O}(\frac{1}{\epsilon})$
STORC [66]	$\mathcal{O}(\kappa^2 d + nd)$	$\mathcal{O}(\log\frac{1}{\epsilon})$
Primal Dual FW (ours)	$\mathcal{O}(ns)$	$\mathcal{O}((1+\kappa/n)\log\frac{1}{\epsilon})$

Table 2.1: Time complexity comparisons on the setting of Corollary 2.5.2. For clear comparison, we refer the per iteration cost as the time complexity of outer iterations.

2.5.1 Extension to the Trace Norm Ball

We also extend our algorithm to matrix optimization over trace norm constraints:

$$\min_{\|X\|_*\leq\lambda, X\in\mathbb{R}^{d\times c}}\left\{\frac{1}{n}\sum_{i=1}^{n}f_i(\boldsymbol{a}_i^\top X)+g(X)\right\}. \tag{2.14}$$

This formulation covers multi-label multi-class problems, matrix completion, affine rank minimization, and phase retrieval problems (see reference therein [20, 5]). Equivalently, we solve the following primal-dual problem:

$$\min_{\|X\|_*\leq\lambda, X\in\mathbb{R}^{d\times c}}\max_{Y\in\mathbb{R}^{n\times c}}\left\{\mathcal{L}(X,Y)\equiv g(X)+\frac{1}{n}\langle AX,Y\rangle-\frac{1}{n}\sum_{i=1}^{n}f_i(\boldsymbol{y}_i)\right\}.$$

Here $\boldsymbol{y}_i$ is the i-th row of the dual matrix Y. For this problem, the *partial update* we enforced on the primal matrix is to keep the update matrix low rank:

$$\tilde{X}\leftarrow\operatorname*{arg\,min}_{\|X\|_*\leq\lambda,\text{rank}(X)\leq s}\left\{\langle\frac{1}{n}Z+\nabla g(X^{(t-1)}),X\rangle+\frac{L}{2}\eta\|X-X^{(t-1)}\|^2\right\}, Z\equiv A^\top Y^{(t-1)}. \tag{2.15}$$

However, an exact solution to (2.15) requires computing the top s left and right singular vectors of the matrix $X^{(t-1)}-\frac{1}{\eta L}(Z+\nabla g(X^{(t-1)})\in\mathbb{R}^{d\times c}$. Therefore we loosely compute an $(\frac{1}{2},\epsilon/2)$-approximation, where ϵ is the target accuracy, based on the following definition:

Algorithm 2 Primal-Dual Block Generalized Frank-Wolfe Method for Trace Norm Ball

1: **Input:** Training data $A \in \mathbb{R}^{n\times d}$, primal and dual step size $\eta, \delta > 0$. Target accuracy ϵ.
2: **Initialize:** $X^{(0)} \leftarrow 0 \in \mathbb{R}^{d\times c}, Y^{(0)} \leftarrow 0 \in \mathbb{R}^{n\times c}, W^{(0)} \equiv AX = 0 \in \mathbb{R}^{n\times c}, Z^{(0)} \equiv A^\top Y = 0 \in \mathbb{R}^{d\times c}$
3: **for** $t = 1, 2, \cdots, T$ **do**
4: Use Frank Wolfe to Update the primal variable:

$$X^{(t)} \leftarrow (1-\eta)X^{(t-1)} + \eta\tilde{X}, \text{ where } \tilde{X} \leftarrow (\frac{1}{2}, \frac{\epsilon}{8})\text{-approximation of Eqn. (2.15).}$$

5: Update W to maintain the value of AX:

$$W^{(t)} \leftarrow (1-\eta)W^{(t-1)} + \eta A\tilde{X} \tag{2.16}$$

6: Consider the potential dual update:

$$\tilde{Y}^{(t)} \leftarrow \arg\max_Y \left\{ \langle W, Y\rangle - f(Y) - \frac{1}{2\delta}\|Y - Y^{(t-1)}\|^2 \right\} \tag{2.17}$$

7: Choose greedily the rows of the dual variable to update: let $I^{(t)}$ be the top k coordinates that maximize

$$\left\|\tilde{Y}_{i,:} - Y^{(t-1)}_{i,:}\right\|_2, i \in [n].$$

Update the dual variable accordingly:

$$Y^{(t)}_{i,:} \leftarrow \begin{cases} \tilde{Y}_{i,:} & \text{if } i \in I^{(t)} \\ Y^{(t-1)}_{i,:} & \text{otherwise.} \end{cases} \tag{2.18}$$

8: Update Z to maintain the value of $A^\top Y$

$$Z^{(t)} \leftarrow Z^{(t-1)} + A^\top(Y^{(t)} - Y^{(t-1)}) \tag{2.19}$$

9: **end for**
10: **Output:** $X^{(T)}, Y^{(T)}$

Definition 2.5.3 (Restated Definition 3.2 in [5]). *Let* $l_t(V) = \langle \nabla_X \mathcal{L}(X^{(t)}, Y^{(t)}), V - X^{(t)}\rangle + \frac{L}{2}\eta\|V - X^{(t)}\|_F^2$ *be the objective function in* (2.15)*, and let* $l_t^* = l_t(\bar{X}^{(t)})$*. Given parameters* $\gamma \geq 0$ *and* $\epsilon \geq 0$*, a feasible solution* V *to* (2.15) *is called* (γ, ϵ)*-approximate if it satisfies* $l(V) \leq (1-\gamma)l_t^* + \epsilon$.

The time dependence on the data size n, c, d, s is $ncs+s^2(n+c)$ [5], and is again independent of d. Meanwhile, the procedures to keep track of $W^{(t)} \equiv AX^{(t)}$ requires complexity of $nds + ncs$, while updating $Y^{(t)}$ requires dck operations. Therefore, by setting $k \leq ns(1/c + 1/d)$, the iteration complexity's dependence on the data size becomes $\mathcal{O}(n(d + c)s)$ operations, instead of $\mathcal{O}(ndc)$ for conducting a full projected gradient step. Recall that s upper bounds the rank of $\bar{X}^{(t)} \leq \min\{d, c\}$.

The trace norm version mostly inherits the convergence guarantees for vector optimization. Refer to the Appendix for details.

Assumption 2.5.1. *We assume the following property on the primal form (2.14):*

- *f_i is $\frac{1}{\beta}$-strongly convex, and satisfies $\frac{1}{\alpha}$-smooth on some convex set (could be $\mathbb{R}^c$) and infinity otherwise.*
- *Data matrix A satisfies $R = \max_{|I|\leq k, I\subset[n]} \sigma^2_{\max}(A_{I,:})$ ($\leq \|A\|_2^2$). Here $\sigma_{\max}(X)$ denotes the largest singular value of X.*
- *g is μ-strongly convex and L-smooth.*

The assumptions also cover smooth hinge loss as well as quadratic loss. With the similar assumptions, the convergence analysis for Algorithm 2 is almost the same as Algorithm 1. The only difference comes from the primal step where approximated update produces some error:

Primal progress: With the primal update rule in Algorithm 2, it satisfies $\mathcal{L}(X^{(t+1)}, Y^{(t)}) - \mathcal{L}(X^{(t)}, Y^{(t)}) \leq -\frac{\mu}{8L}\Delta_p^{(t)} + \frac{\epsilon}{16}$. (See Lemma A.1.7.) With no much modification in the proof, we are able to derive similar convergence guarantees for the trace norm ball.

Theorem 2.5.4. *Given a function $\frac{1}{n}\sum_{i=1}^n f_i(\boldsymbol{a}_i^\top X) + g(X)$ that satisfies Assumption 2.5.1. Setting $s \geq$rank$(\bar{X}^{(t)})$, and learning rate $\eta = \frac{\mu}{2L}, \delta \leq \frac{1}{k}(\frac{L}{\mu n\beta} + \frac{5\beta R}{2\alpha\mu n^2}(1 + 8\frac{L}{\mu}))^{-1}$, the duality gap $\Delta^{(t)}$*

generated by Algorithm 2 satisfies $\Delta^{(t)} \leq \frac{k\delta}{k\delta+8\beta n}\Delta^{(t-1)} + \frac{\epsilon}{16}$. *Therefore it takes* $\mathcal{O}(\frac{L}{\alpha}(1+\frac{\beta}{\alpha}\frac{R\beta}{n\mu})\log\frac{1}{\epsilon})$ *iterations to achieve* ϵ *error.*

We also provide a brief analysis on the difficulty to extend our algorithm to polytope-type constraints in the Appendix A.1.8.

2.6 Experiments

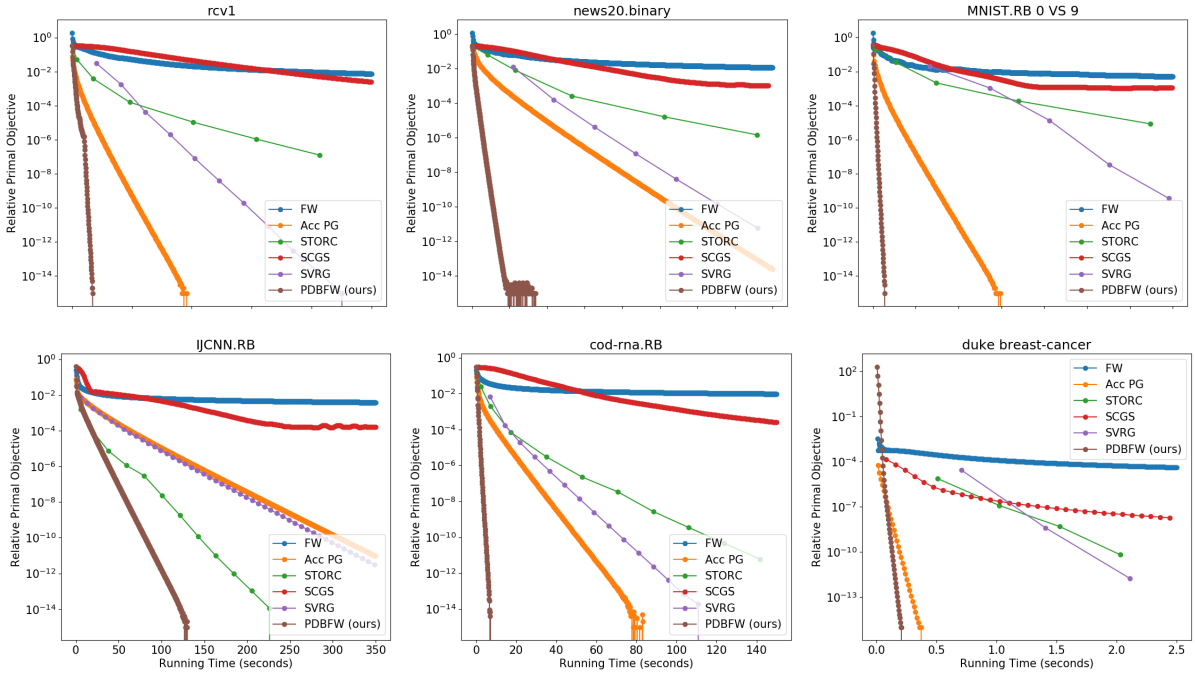


Figure 2.1: **Convergence result comparison of different algorithms on smoothed hinge loss.** For six different datasets, we show the decrease of relative primal objective: $(P(\boldsymbol{x}^{(t)}) - P^*)/P^*$ over CPU time. Our algorithm (brown) achieves around 10 times speedup over all other methods except for the smallest dataset duke.

We evaluate the Primal-Dual Block Generalized Frank-Wolfe algorithm by its performance

on binary classification with smoothed hinge loss[3]. We refer the readers to Appendix A.1.6 for details about smoothed hinge loss.

We compare the proposed algorithm against five benchmark algorithms: (1) Accelerated Projected Gradient Descent (Acc PG) (2) Frank-Wolfe algorithm (FW) (3) Stochastic Variance Reduced Gradient (SVRG) [74] (4) Stochastic Conditional Gradient Sliding (SCGS) [87] and (5) Stochastic Variance-Reduced Conditional Gradient Sliding (STORC) [66]. We presented the time complexity for each algorithm in Table 2.1. Three of them (FW, SCGS, STORC) are projection-free algorithms, and the other two (Acc PG, SVRG) are projection-based algorithms. Algorithms are implemented in C++, with the Eigen linear algebra library [64].

The six datasets used here are summarized in Table 2.2. All of them can be found in LIBSVM datasets [22]. We augment the features of MNIST, ijcnn, and cob-rna by random binning [138], which is a standard technique for kernel approximation. Data is normalized. We set the ℓ_1 constraint to be 300 and the ℓ_2 regularize parameter to $10/n$ to achieve reasonable prediction accuracy. We refer the readers to the Appendix A.3.1 for results of other choice of parameters. These datasets have various scale of features, samples, and solution sparsity ratio.

The results are shown in Fig 2.1. To focus on the convergence property, we show the decrease of loss function instead of prediction accuracy. From Fig 2.1, our proposed algorithm consistently outperforms the benchmark algorithms. The winning margin is roughly proportional to the solution sparsity ratio, which is consistent with our theory.

We also implement Algorithm 2 for trace norm ball and compare it with some prior work in the Appendix A.3.2, especially Block FW [5]. We generated synthetic data with optimal solutions

[3]The codes to reproduce our results could be found in https://github.com/CarlsonZhuo/primal_dual_frank_wolfe.

of different ranks, and show that our proposal is consistently faster than others.

Dataset Name	# Features	# Samples	# Non-Zero	Solution Sparsity (Ratio)
duke breast-cancer [22]	7,129	44	313,676	423 (5.9%)
rcv1 [22]	47,236	20,242	1,498,952	1,169 (2.5%)
news20.binary [22]	1,355,191	19,996	9,097,916	1,365 (0.1%)
MNIST.RB 0 VS 9 [22, 138]	894,499	11,872	1,187,200	8,450 (0.9%)
ijcnn.RB [22, 138]	58,699	49,990	14,997,000	715 (1.2%)
cob-rna.RB [22, 138]	81,398	59,535	5,953,500	958 (1.2%)

Table 2.2: Summary of the properties of the datasets.

2.7 Conclusion

In this paper we consider a class of problems whose solutions enjoy some simple structure induced by the constraints. We argue that the class of algorithms that conduct sparse updates is able to exploit the simple structure. Specifically, we propose a FW type algorithm and greedy coordinate descent to reduce time cost for each update remarkably while attaining linear convergence. For a class of ERM problems, our running time depends on the sparsity/rank of the optimal solutions rather than the ambient feature dimension. Our empirical studies verify the improved performance compared to various state-of-the-art algorithms.

Chapter 3

Convex-Concave Games: On Last-Iterate Convergence

In a recent series of papers it has been established that variants of Gradient Descent/Ascent and Mirror Descent exhibit last iterate convergence in convex-concave zero-sum games. Specifically, [39, 103] show last iterate convergence of the so called "Optimistic Gradient Descent/Ascent" for the case of *unconstrained* min-max optimization. Moreover, in [113] the authors show that Mirror Descent with an extra gradient step displays last iterate convergence for convex-concave problems (both constrained and unconstrained), though their algorithm does not follow the online learning framework; it uses extra information rather than *only* the history to compute the next iteration. In this work, we show that "Optimistic Multiplicative-Weights Update (OMWU)" which follows the no-regret online learning framework, exhibits last iterate convergence locally for convex-concave games, generalizing the results of [40] where last iterate convergence of OMWU was shown only for the *bilinear case*. We complement our results with experiments that indicate fast convergence of the method.[1]

3.1 Introduction

In classic (normal form) zero-sum games, one has to compute two probability vectors $\vec{x}^* \in \Delta_n, \vec{y}^* \in \Delta_m$ [2] that consist an equilibrium of the following problem

$$\min_{\vec{x}\in\Delta_n} \max_{\vec{y}\in\Delta_m} \vec{x}^\top A\vec{y}, \tag{3.1}$$

where A is $n \times m$ real matrix (called payoff matrix). Here $\vec{x}^\top A\vec{y}$ represents the payment of the $\vec{x}$ player to the $\vec{y}$ player under choices of strategies by the two players and is a *bilinear* function.

[1] This work is based on the following ArXiv papers:
Qi Lei, Sai Ganesh Nagarajan, Ioannis Panageas, Xiao Wang. "Last iterate convergence in no-regret learning: constrained min-max optimization for convex-concave landscapes", arXiv preprint arXiv:2002.06768 [94]

[2] Δ_n denotes the simplex of size n.

Arguably, one of the most celebrated theorems and a founding stone in Game Theory, is the minimax theorem by Von Neumann [160]. It states that

$$\min_{\vec{x}\in\Delta_n}\max_{\vec{y}\in\Delta_m} f(\vec{x},\vec{y}) = \max_{\vec{y}\in\Delta_m}\min_{\vec{x}\in\Delta_n} f(\vec{x},\vec{y}), \tag{3.2}$$

where $f : \Delta_n \times \Delta_m \to \mathbb{R}$ is convex in $\vec{x}$, concave in $\vec{y}$. The aforementioned result holds for any convex compact sets $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}^m$. The min-max theorem reassures us that an equilibrium always exists in the bilinear game (3.1) or its convex-concave analogue (again $f(\vec{x},\vec{y})$ is interpreted as the payment of the $\vec{x}$ player to the $\vec{y}$ player). An equilibrium is a pair of randomized strategies $(\vec{x}^*,\vec{y}^*)$ such that neither player can improve their payoff by unilaterally changing their distribution.

Soon after the appearance of the minimax theorem, research was focused on dynamics for solving min-max optimization problems by having the min and max players of (3.1) run a simple online learning procedure. In the online learning framework, at time t, each player chooses a probability distribution ($\vec{x}^t, \vec{y}^t$ respectively) simultaneously depending *only* on the past choices of both players (i.e., $\vec{x}^1, ..., \vec{x}^{t-1}, \vec{y}^1, ..., \vec{y}^{t-1}$) and experiences payoff that depends on choices $\vec{x}^t, \vec{y}^t$.

An early method, proposed by Brown [18] and analyzed by Robinson [142], was fictitious play. Later on, researchers discover several learning robust algorithms converging to minimax equilibrium at faster rates, see [21]. This class of learning algorithms, are the so-called "no-regret" and include Multiplicative Weights Update method [11] and Follow the regularized leader.

3.1.1 Average Iterate Convergence

Despite the rich literature on no-regret learning, most of the known results have the feature that min-max equilibrium is shown to be attained only by the time *average*. This means that the trajectory of a no-regret learning method $(\vec{x}^t, \vec{y}^t)$ has the property that $\frac{1}{t}\sum_{\tau\le t}(\vec{x}^\tau)^\top A\vec{y}^\tau$ converges

to the equilibrium of (3.1), as $t \to \infty$. Unfortunately that does not mean that the last iterate $(\vec{x}^t, \vec{y}^t)$ converges to an equilibrium, it commonly diverges or cycles. One such example is the well-known Multiplicative Weights Update Algorithm, the time average of which is known to converge to an equilibrium, but the actual trajectory cycles towards the boundary of the simplex ([14]). This is even true for the vanilla Gradient Descent/Ascent, where one can show for even bilinear landscapes (unconstrained case) last iterate fails to converge [39].

Motivated by the training of Generative Adversarial Networks (GANs), the last couple of years researchers have focused on designing and analyzing procedures that exhibit *last iterate* convergence (or pointwise convergence) for zero-sum games. This is crucial for training GANs, the landscapes of which are typically non-convex non-concave and averaging now as before does not give much guarantees (e.g., note that Jensen's inequality is not applicable anymore). In [39, 103] the authors show that a variant of Gradient Descent/Ascent, called Optimistic Gradient Descent/Ascent has last iterate convergence for the case of bilinear functions $\vec{x}^\top A\vec{y}$ where $\vec{x} \in \mathbb{R}^n$ and $\vec{y} \in \mathbb{R}^m$ (this is called the unconstrained case, since there are no restrictions on the vectors). Later on, [40] generalized the above result with simplex constraints, where the online method that the authors analyzed was Optimistic Multiplicative Weights Update. In [113], it is shown that Mirror Descent with extra gradient computation converges pointwise for a class of zero-sum games that includes the convex-concave setting (with arbitrary constraints), though their algorithm does not fit in the online no-regret framework since it uses information twice about the payoffs before it iterates. Last but not least there have appeared other works that show pointwise convergence for other settings (see [131, 41] and [1] and references therein) to stationary points (but not local equilibrium solutions).

3.1.2 Main Results

In this work, we focus on the min-max optimization problem

$$\min_{\vec{x}\in\Delta_n}\max_{\vec{y}\in\Delta_m} f(\vec{x},\vec{y}), \tag{3.3}$$

where f is a convex-concave function (convex in $\vec{x}$, concave in $\vec{y}$). We analyze the no-regret online algorithm Optimistic Multiplicative Weights Update (OMWU). OMWU is an instantiation of the Optimistic Follow the Regularized Leader (OFTRL) method with entropy as a regularizer (for both players, see Preliminaries section for the definition of OMWU).

We prove that OMWU exhibits local last iterate convergence, generalizing the result of [40] and proving an open question of [154] (for convex-concave games). Formally, our main theorem is stated below:

Theorem 3.1.1 (Last iterate convergence of OMWU)**.** *Let $f : \Delta_n \times \Delta_m \to \mathbb{R}$ be a twice differentiable function $f(\vec{x},\vec{y})$ that is convex in $\vec{x}$ and concave in $\vec{y}$. Assume that there exists an equilibrium $(\vec{x}^*,\vec{y}^*)$ that satisfies the KKT conditions with strict inequalities (see (3.4)). It holds that for sufficiently small stepsize, there exists a neighborhood $U \subseteq \Delta_n \times \Delta_m$ of $(\vec{x}^*,\vec{y}^*)$ such that for all for all initial conditions $(\vec{x}^0,\vec{y}^0),(\vec{x}^1,\vec{y}^1) \in U$, OMWU exhibits last iterate (pointwise) convergence, i.e.,*

$$\lim_{t\to\infty}(\vec{x}^t,\vec{y}^t) = (\vec{x}^*,\vec{y}^*),$$

where $(\vec{x}^t,\vec{y}^t)$ denotes the t-th iterate of OMWU.

Moreover, we complement our theoretical findings with experimental analysis of the procedure. The experiments on KL-divergence indicate that the results should hold globally.

3.1.3 Structure and Technical Overview

We present the structure of the paper and a brief technical overview.

Section 2 provides necessary definitions, the explicit form of OMWU derived from OFTRL with entropy regularizer, and some existing results on dynamical systems.

Section 3 is the main technical part, i.e, the computation and spectral analysis of the Jacobian matrix of OMWU dynamics. The stability analysis, the understanding of the local behavior and the local convergence guarantees of OMWU rely on the spectral analysis of the computed Jacobian matrix. The techniques for bilinear games (as in [40]) are no longer valid in convex-concave games. Allow us to explain the differences from [40]. In general, one cannot expect a trivial generalization from linear to non-linear scenarios. The properties of bilinear games are fundamentally different from that of convex-concave games, and this makes the analysis much more challenging in the latter. The key result of spectral analysis in [40] is in a lemma (Lemma B.6) which states that a skew symmetric[3] has imaginary eigenvalues. Skew symmetric matrices appear since in bilinear cases there are terms that are linear in $\vec{x}$ and linear in $\vec{y}$ but no higher order terms in $\vec{x}$ or $\vec{y}$. However, the skew symmetry has no place in the case of convex-concave landscapes and the Jacobian matrix of OMWU is far more complicated. One key technique to overcome the lack of skew symmetry is the use of Ky Fan inequality [120] which states that the sequence of the eigenvalues of $\frac{1}{2}(W + W^\top)$ majorizes the real part of the sequence of the eigenvalues of W for any square matrix W (see Lemma 3.1).

Section 4 focuses on numerical experiments to understand how the problem size and the choice of learning rate affect the performance of our algorithm. We observe that our algorithm is

[3] A is skew symmetric if $A^\top = -A$.

able to achieve global convergence invariant to the choice of learning rate, random initialization or problem size. As comparison, the latest popularized (projected) optimistic gradient descent ascent is much more sensitivity to the choice of hyperparameter. Due to space constraint, the detailed calculation of the Jacobian matrix (general form and at fixed point) of OMWU are left in Appendix.

Notation The boldface $\vec{x}$ and $\vec{y}$ denote the vectors in Δ_n and Δ_m. $\vec{x}^t$ denotes the t-th iterate of the dynamical system. The letter J denote the Jacobian matrix. $\vec{I}$, $\vec{0}$ and $\vec{1}$ are preserved for the identity, zero matrix and the vector with all the entries equal to 1. The support of $\vec{x}$ is the set of indices of x_i such that $x_i \neq 0$, denoted by Supp($\vec{x}$). $(\vec{x}^*, \vec{y}^*)$ denotes the optimal solution for minimax problem. $[n]$ denote the set of integers $\{1, ..., n\}$.

3.2 Preliminaries

In this section, we present some background that will be used later.

3.2.1 Equilibria for Constrained Minimax

From Von Neumann's minimax theorem, one can conclude that the problem $\min_{\vec{x}\in\Delta_n} \max_{\vec{y}\in\Delta} f(\vec{x}, \vec{y})$ has always an equilibrium $(\vec{x}^*, \vec{y}^*)$ with $f(\vec{x}^*, \vec{y}^*)$ be unique. Moreover from KKT conditions (as long as f is twice differentiable), such an equilibrium must satisfy the following ($\vec{x}^*$ is a local minimum for fixed $\vec{y} = \vec{y}^*$ and $\vec{y}^*$ is a local maximum for fixed $\vec{x} = \vec{x}^*$):

Definition 3.2.1 (KKT conditions)**.** *Formally, it holds*

$$\begin{aligned}
&\vec{x}^* \in \Delta_n \\
&x_i^* > 0 \Rightarrow \tfrac{\partial f}{\partial x_i}(\vec{x}^*, \vec{y}^*) = \textstyle\sum_{j=1}^n x_j^* \tfrac{\partial f}{\partial x_j}(\vec{x}^*, \vec{y}^*) \\
&x_i^* = 0 \Rightarrow \tfrac{\partial f}{\partial x_i}(\vec{x}^*, \vec{y}^*) \geq \textstyle\sum_{j=1}^n x_j^* \tfrac{\partial f}{\partial x_j}(\vec{x}^*, \vec{y}^*) \\
&\textit{for player } \vec{x}, \\
&\vec{y}^* \in \Delta_m \\
&y_i^* > 0 \Rightarrow \tfrac{\partial f}{\partial y_i}(\vec{x}^*, \vec{y}^*) = \textstyle\sum_{j=1}^m y_j^* \tfrac{\partial f}{\partial y_j}(\vec{x}^*, \vec{y}^*) \\
&y_i^* = 0 \Rightarrow \tfrac{\partial f}{\partial y_i}(\vec{x}^*, \vec{y}^*) \leq \textstyle\sum_{j=1}^m y_j^* \tfrac{\partial f}{\partial y_j}(\vec{x}^*, \vec{y}^*) \\
&\textit{for player } \vec{y}.
\end{aligned} \tag{3.4}$$

Remark 3.2.1 (No degeneracies)**.** *For the rest of the paper we assume no degeneracies, i.e., the last inequalities hold strictly (in the case a strategy is played with zero probability for each player). Moreover, it is easy to see that since f is convex concave and twice differentiable, then $\nabla^2_{\vec{x}\vec{x}} f$ (part of the Hessian that involves $\vec{x}$ variables) is positive semi-definite and $\nabla^2_{\vec{y}\vec{y}} f$ (part of the Hessian that involves $\vec{y}$ variables) is negative semi-definite.*

3.2.2 Optimistic Multiplicative Weights Update

The equations of Optimistic Follow-the-Regularized-Leader (OFTRL) applied to a problem $\min_{\vec{x}\in\mathcal{X}} \max_{\vec{y}\in\mathcal{Y}} f(\vec{x}, \vec{y})$ with regularizers (strongly convex functions) $h_1(\vec{x}), h_2(\vec{y})$ (for player $\vec{x}, \vec{y}$ respectively) and $\mathcal{X} \subset \mathbb{R}^n, \mathcal{Y} \subset \mathbb{R}^m$ is given below (see [39]):

$$\vec{x}^{t+1} = \arg\min_{\vec{x}\in\mathcal{X}} \{\eta \sum_{s=1}^{t} \vec{x}^\top \nabla_{\vec{x}} f(\vec{x}^s, \vec{y}^s) + \underbrace{\eta \vec{x}^\top \nabla_{\vec{x}} f(\vec{x}^t, \vec{y}^t)}_{\text{optimistic term}} + h_1(\vec{x})\}$$

$$\vec{y}^{t+1} = \arg\max_{\vec{y}\in\mathcal{Y}} \{\eta \sum_{s=1}^{t} \vec{y}^\top \nabla_{\vec{y}} f(\vec{x}^s, \vec{y}^s) + \underbrace{\eta \vec{y}^\top \nabla_{\vec{y}} f(\vec{x}^t, \vec{y}^t)}_{\text{optimistic term}} - h_2(\vec{y})\}.$$

η is called the *stepsize* of the online algorithm. OFTRL is uniquely defined if f is convex-concave and domains $\mathcal{X}$ and $\mathcal{Y}$ are convex. For simplex constraints and entropy regularizers, i.e., $h_1(\vec{x}) =$

$\sum_i x_i \ln x_i$, $h_2(\vec{y}) = \sum_i y_i \ln y_i$, we can solve for the explicit form of OFTRL using KKT conditions, the update rule is the Optimistic Multiplicative Weights Update (OMWU) and is described as follows:

$$x_i^{t+1} = x_i^t \frac{e^{-2\eta \frac{\partial f}{\partial x_i}(\vec{x}^t, \vec{y}^t) + \eta \frac{\partial f}{\partial x_i}(\vec{x}^{t-1}, \vec{y}^{t-1})}}{\sum_k x_k^t e^{-2\eta \frac{\partial f}{\partial x_k}(\vec{x}^t, \vec{y}^t) + \eta \frac{\partial f}{\partial x_k}(\vec{x}^{t-1}, \vec{y}^{t-1})}}$$
$$\text{for all } i \in [n],$$
$$y_i^{t+1} = y_i^t \frac{e^{2\eta \frac{\partial f}{\partial y_i}(\vec{x}^t, \vec{y}^t) - \eta \frac{\partial f}{\partial y_i}(\vec{x}^{t-1}, \vec{y}^{t-1})}}{\sum_k y_k^t e^{2\eta \frac{\partial f}{\partial y_j}(\vec{x}^t, \vec{y}^t) - \eta \frac{\partial f}{\partial y_k}(\vec{x}^{t-1}, \vec{y}^{t-1})}}$$
$$\text{for all } i \in [m].$$

3.2.3 Fundamentals of Dynamical Systems

We conclude Preliminaries section with some basic facts from dynamical systems.

Definition 3.2.2. *A recurrence relation of the form $\vec{x}^{t+1} = w(\vec{x}^t)$ is a discrete time dynamical system, with update rule $w : \mathcal{S} \to \mathcal{S}$ where $\mathcal{S}$ is a subset of $\mathbb{R}^k$ for some positive integer k. The point $\vec{z} \in \mathcal{S}$ is called a* fixed point *if $w(\vec{z}) = \vec{z}$.*

Remark 3.2.2. *Using KKT conditions (3.4), it is not hard to observe that an equilibrium point $(\vec{x}^*, \vec{y}^*)$ must be a fixed point of the OMWU algorithm, i.e., if $(\vec{x}^t, \vec{y}^t) = (\vec{x}^{t-1}, \vec{y}^{t-1}) = (\vec{x}^*, \vec{y}^*)$ then $(\vec{x}^{t+1}, \vec{y}^{t+1}) = (\vec{x}^*, \vec{y}^*)$.*

Proposition 3.2.3 ([54])**.** *Assume that w is a differentiable function and the Jacobian of the update rule w at a fixed point $\vec{z}^*$ has spectral radius less than one. It holds that there exists a neighborhood U around $\vec{z}^*$ such that for all $\vec{z}^0 \in U$, the dynamics $\vec{z}^{t+1} = w(\vec{z}^t)$ converges to $\vec{z}^*$, i.e. $\lim_{n\to\infty} w^n(\vec{z}^0) = \vec{z}^*$ [4]. w is called a contraction mapping in U.*

[4] w^n denotes the composition of w with itself n times.

Note that we will make use of Proposition 3.2.3 to prove our Theorem 3.1.1 (by proving that the Jacobian of the update rule of OMWU has spectral radius less than one).

3.3 Last iterate convergence of OMWU

In this section, we prove that OMWU converges pointwise (exhibits last iterate convergence) if the initializations $(\vec{x}^0, \vec{y}^0), (\vec{x}^1, \vec{y}^1)$ belong in a neighborhood U of the equilibrium $(\vec{x}^*, \vec{y}^*)$.

3.3.1 Dynamical System of OMWU

We first express OMWU algorithm as a dynamical system so that we can use Proposition 3.2.3. The idea (similar to [40]) is to lift the space to consist of four components $(\vec{x}, \vec{y}, \vec{z}, \vec{w}$, in such a way we can include the history (current and previous step, see Section 3.2.2 for the equations). First, we provide the update rule $g : \Delta_n \times \Delta_m \times \Delta_n \times \Delta_m \to \Delta_n \times \Delta_m \times \Delta_n \times \Delta_m$ of the lifted dynamical system and is given by

$$g(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = (g_1, g_2, g_3, g_4)$$

where $g_i = g_i(\vec{x}, \vec{y}, \vec{z}, \vec{w})$ for $i \in [4]$ are defined as follows:

$$g_{1,i}(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = x_i \frac{e^{-2\eta \frac{\partial f}{\partial x_i}(\vec{x},\vec{y}) + \eta \frac{\partial f}{\partial z_i}(\vec{z},\vec{w})}}{\sum_k x_k e^{-2\eta \frac{\partial f}{\partial x_k}(\vec{x},\vec{y}) + \eta \frac{\partial f}{\partial z_k}(\vec{z},\vec{w})}}, i \in [n] \tag{3.5}$$

$$g_{2,i}(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = y_i \frac{e^{2\eta \frac{\partial f}{\partial y_i}(\vec{x},\vec{y}) - \eta \frac{\partial f}{\partial w_i}(\vec{z},\vec{w})}}{\sum_k y_k e^{2\eta \frac{\partial f}{\partial y_k}(\vec{x},\vec{y}) - \eta \frac{\partial f}{\partial w_k}(\vec{z},\vec{w})}}, i \in [m] \tag{3.6}$$

$$g_3(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = \vec{x} \quad \text{or} \quad g_{3,i}(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = x_i, \ i \in [n] \tag{3.7}$$

$$g_4(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = \vec{y} \quad \text{or} \quad g_{4,i}(\vec{x}, \vec{y}, \vec{z}, \vec{w}) = y_i, \ i \in [m]. \tag{3.8}$$

Then the dynamical system of OMWU can be written in compact form as

$$(\vec{x}_{t+1}, \vec{y}_{t+1}, \vec{x}_t, \vec{y}_t) = g(\vec{x}_t, \vec{y}_t, \vec{x}_{t-1}, \vec{y}_{t-1}).$$

In what follows, we will perform spectral analysis on the Jacobian of the function g, computed at the fixed point $(\vec{x}^*, \vec{y}^*)$. Since g has been lifted, the fixed point we analyze is $(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*)$ (see Remark 3.2.2). By showing that the spectral radius is less than one, our Theorem 3.1.1 follows by Proposition 3.2.3. The computations of the Jacobian of g are deferred to the supplementary material.

3.3.2 Spectral Analysis

Let $(\vec{x}^*, \vec{y}^*)$ be the equilibrium of min-max problem (3.2). Assume $i \notin \text{Supp}(\vec{x}^*)$, i.e., $x_i^* = 0$ then (see equations at the supplementary material, section A)

$$\frac{\partial g_{1,i}}{\partial x_i}(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*) = \frac{e^{-\eta \frac{\partial f}{\partial x_i}(\vec{x}^*, \vec{y}^*)}}{\sum_{t=1}^n x_t^* e^{-\eta \frac{\partial f}{\partial x_t}(\vec{x}^*, \vec{y}^*)}}$$

and all other partial derivatives of $g_{1,i}$ are zero, thus $\frac{e^{-\eta \frac{\partial f}{\partial x_i}(\vec{x}^*, \vec{y}^*)}}{\sum_{t=1}^n x_t^* e^{-\eta \frac{\partial f}{\partial x_t}(\vec{x}^*, \vec{y}^*)}}$ is an eigenvalue of the Jacobian computed at $(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*)$. This is true because the row of the Jacobian that corresponds to $g_{1,i}$ has zeros everywhere but the diagonal entry. Moreover because of the degeneracy assumption of KKT conditions (see Remark 3.2.1), it holds that

$$\frac{e^{-\eta \frac{\partial f}{\partial x_i}(\vec{x}^*, \vec{y}^*)}}{\sum_{t=1}^n x_t^* e^{-\eta \frac{\partial f}{\partial x_t}(\vec{x}^*, \vec{y}^*)}} < 1.$$

Similarly, it holds for $j \notin \text{Supp}(\vec{y}^*)$ that

$$\frac{\partial g_{2,j}}{\partial y_j}(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*) = \frac{e^{\eta \frac{\partial f}{\partial y_j}(\vec{x}^*, \vec{y}^*)}}{\sum_{t=1}^m y_t^* e^{\eta \frac{\partial f}{\partial y_t}(\vec{x}^*, \vec{y}^*)}} < 1$$

(again by Remark 3.2.1) and all other partial derivatives of $g_{2,j}$ are zero, therefore $\frac{e^{\eta \frac{\partial f}{\partial y_j}(\vec{x}^*, \vec{y}^*)}}{\sum_{t=1}^{m} y_t^* e^{\eta \frac{\partial f}{\partial y_t}(\vec{x}^*, \vec{y}^*)}}$ is an eigenvalue of the Jacobian computed at $(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*)$.

We focus on the submatrix of the Jacobian of g computed at $(\vec{x}^*, \vec{y}^*, \vec{x}^*, \vec{y}^*)$ that corresponds to the non-zero probabilities of $\vec{x}^*$ and $\vec{y}^*$. We denote $D_{\vec{x}^*}$ to be the diagonal matrix of size $|\text{Supp}(\vec{x}^*)| \times |\text{Supp}(\vec{x}^*)|$ that has on the diagonal the nonzero entries of $\vec{x}^*$ and similarly we define $D_{\vec{y}^*}$ of size $|\text{Supp}(\vec{y}^*)| \times |\text{Supp}(\vec{y}^*)|$. For convenience, let us denote $k_x := |\text{Supp}(\vec{x}^*)|$ and $k_y := |\text{Supp}(\vec{y}^*)|$. The Jacobian submatrix is the following

$$J = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ \vec{I}_{k_x \times k_x} & \vec{0}_{k_x \times k_y} & \vec{0}_{k_x \times k_x} & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & \vec{I}_{k_y \times k_y} & \vec{0}_{k_y \times k_x} & \vec{0}_{k_y \times k_y} \end{bmatrix}$$

where

$$\begin{aligned}
A_{11} &= \vec{I}_{k_x \times k_x} - D_{\vec{x}^*} \vec{1}_{k_x} \vec{1}_{k_x}^\top - 2\eta D_{\vec{x}^*} (\vec{I}_{k_x \times k_x} - \vec{1}_{k_x} \vec{x}^{*\top}) \nabla^2_{\vec{x}\vec{x}} f \\
A_{12} &= -2\eta D_{\vec{x}^*} (\vec{I}_{k_x \times k_x} - \vec{1}_{k_x} \vec{x}^{*\top}) \nabla^2_{\vec{x}\vec{y}} f \\
A_{13} &= \eta D_{\vec{x}^*} (\vec{I}_{k_x \times k_x} - \vec{1}_{k_x} \vec{x}^{*\top}) \nabla^2_{\vec{x}\vec{x}} f \\
A_{14} &= \eta D_{\vec{x}^*} (\vec{I}_{k_x \times k_x} - \vec{1}_{k_x} \vec{x}^{*\top}) \nabla^2_{\vec{x}\vec{y}} f \\
A_{21} &= 2\eta D_{\vec{y}^*} (\vec{I}_{k_y \times k_y} - \vec{1}_{k_y} \vec{y}^{*\top}) \nabla^2_{\vec{y}\vec{x}} f \\
A_{22} &= \vec{I}_{k_y \times k_y} - D_{\vec{y}^*} \vec{1}_{k_y} \vec{1}_{k_y}^\top + 2\eta D_{\vec{y}^*} (\vec{I}_{k_y \times k_y} - \vec{1}_{k_y} \vec{y}^{*\top}) \nabla^2_{\vec{y}\vec{y}} f \\
A_{23} &= -\eta D_{\vec{y}^*} (\vec{I}_{k_y \times k_y} - \vec{1}_{k_y} \vec{y}^{*\top}) \nabla^2_{\vec{y}\vec{x}} f \\
A_{24} &= -\eta D_{\vec{y}^*} (\vec{I}_{k_y \times k_y} - \vec{1}_{k_y} \vec{y}^{*\top}) \nabla^2_{\vec{y}\vec{y}} f.
\end{aligned} \tag{3.9}$$

We note that $\vec{I}, \vec{0}$ capture the identity matrix and the all zeros matrix respectively (the appropriate size is indicated as a subscript). The vectors $(\vec{1}_{k_x}, \vec{0}_{k_y}, \vec{0}_{k_x}, \vec{0}_{k_y})$ and $(\vec{0}_{k_x}, \vec{1}_{k_y}, \vec{0}_{k_x}, \vec{0}_{k_y})$ are left eigenvectors with eigenvalue zero for the above matrix. Hence, any right eigenvector $(\vec{v_x}, \vec{v_y}, \vec{v_z}, \vec{v_w})$

should satisfy the conditions $\vec{1}^\top \vec{v_x} = 0$ and $\vec{1}^\top \vec{v_y} = 0$. Thus, every non-zero eigenvalue of the above matrix is also a non-zero eigenvalue of the matrix below:

$$J_{\text{new}} = \begin{bmatrix} B_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & B_{22} & A_{23} & A_{24} \\ \vec{I}_{k_x \times k_x} & \vec{0}_{k_x \times k_y} & \vec{0}_{k_x \times k_x} & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & \vec{I}_{k_y \times k_y} & \vec{0}_{k_y \times k_x} & \vec{0}_{k_y \times k_y} \end{bmatrix}$$

where

$$B_{11} = \vec{I}_{k_x \times k_x} - 2\eta D_{\vec{x}^*}(\vec{I}_{k_x \times k_x} - \vec{1}_{k_x}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}} f,$$

$$B_{22} = \vec{I}_{k_y \times k_y} + 2\eta D_{\vec{y}^*}(\vec{I}_{k_y \times k_y} - \vec{1}_{k_y}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}} f.$$

The characteristic polynomial of J_{new} is obtained by finding $\det(J_{\text{new}} - \lambda \vec{I})$. One can perform row/column operations on J_{new} to calculate this determinant, which gives us the following relation:

$$\det(J_{\text{new}} - \lambda \vec{I}_{2k_x \times 2k_y}) = (1-2\lambda)^{(k_x+k_y)}\, q\left(\frac{\lambda(\lambda-1)}{2\lambda-1}\right)$$

where $q(\lambda)$ is the characteristic polynomial of the following matrix

$$J_{\text{small}} = \begin{bmatrix} B_{11} - \vec{I}_{k_x \times k_x} & A_{12} \\ A_{21} & B_{22} - \vec{I}_{k_y \times k_y}, \end{bmatrix}$$

and $B_{11}, B_{12}, A_{12}, A_{21}$ are the aforementioned sub-matrices. Notice that J_{small} can be written as

$$J_{\text{small}} = 2\eta \begin{bmatrix} -(D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix} H$$

where,

$$H = \begin{bmatrix} \nabla^2_{\vec{x}\vec{x}} f & \nabla^2_{\vec{x}\vec{y}} f \\ \nabla^2_{\vec{y}\vec{x}} f & \nabla^2_{\vec{y}\vec{y}} f \end{bmatrix}$$

Notice here that H is the Hessian matrix evaluated at the fixed point $(\vec{x}^*, \vec{y}^*)$, and is the appropriate sub-matrix restricted to the support of $|\text{Supp}(\vec{y}^*)|$ and $|\text{Supp}(\vec{x}^*)|$. Although, the Hessian matrix is symmetric, we would like to work with the following representation of J_{small}:

$$J_{\text{small}} = 2\eta \begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix} H^-$$

where,

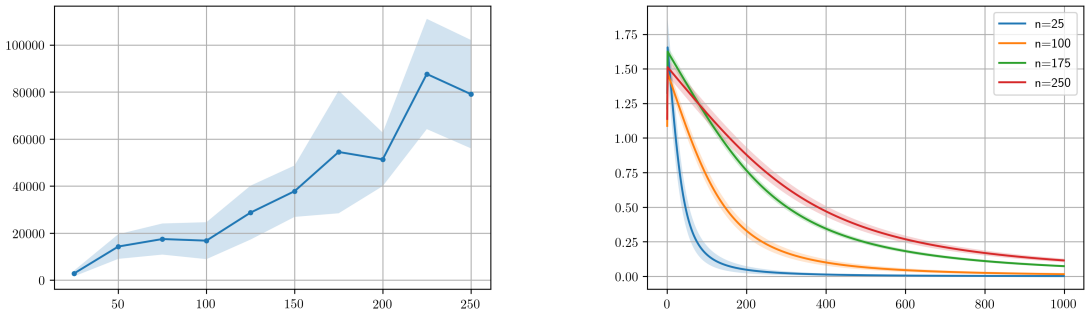
$$H^{-} = \begin{bmatrix} -\nabla^2_{\vec{x}\vec{x}} f & -\nabla^2_{\vec{x}\vec{y}} f \\ \nabla^2_{\vec{y}\vec{x}} f & \nabla^2_{\vec{y}\vec{y}} f \end{bmatrix}$$

Let us denote any non-zero eigenvalue of J_{small} by ϵ which may be a complex number. Thus ϵ is where $q(\cdot)$ vanishes and hence the eigenvalue of J_{new} must satisfy the relation

$$\frac{\lambda(\lambda-1)}{2\lambda-1} = \epsilon$$

We are to now show that the magnitude of any eigenvalue of J_{new} is strictly less than 1, i.e, $|\lambda| < 1$. Trivially, $\lambda = \frac{1}{2}$ satisfies the above condition. Thus we need to show that the magnitude of λ where $q(\cdot)$ vanishes is strictly less than 1. The remainder of the proof proceeds by showing the following two lemmas:

Lemma 3.3.1 (Real part non-positive)**.** *Let λ be an eigenvalue of matrix J_{small}. It holds that $Re(\lambda) \leq 0$.*



(a) #iterations vs size of n

(b) l_1 error vs #iterations

Figure 3.1: *Convergence of OMWU vs different sizes of the problem.* For Figure (a), x-axis is n and y-axis is the number of iterations to reach convergence for Eqn. (3.14). In Figure (b) we choose four cases of n to illustrate how l_1 error of the problem decreases with the number of iterations.

Proof. Assume that $\lambda \neq 0$. All the non-zero eigenvalues of matrix J_{small} coincide with the eigenvalues of the matrix

$$R := \begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix}^{1/2} \times H^- \times \begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix}^{1/2}.$$

This is well-defined since

$$\begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix}$$

is positive semi-definite. Moreover, we use KyFan inequalities which state that the sequence (in decreasing order) of the eigenvalues of $\frac{1}{2}(W + W^\top)$ majorizes the real part of the sequence of the eigenvalues of W for any square matrix W (see [120], page 4). We conclude that for any eigenvalue λ of R, it holds that $\text{Re}(\lambda)$ is at most the maximum eigenvalue of $\frac{1}{2}(R + R^\top)$. Observe now that

$$R + R^\top := \begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix}^{1/2} \times$$

$$(H^- + H^{-\top}) \times \begin{bmatrix} (D_{\vec{x}^*} - \vec{x}^*\vec{x}^{*\top}) & \vec{0}_{k_x \times k_y} \\ \vec{0}_{k_y \times k_x} & (D_{\vec{y}^*} - \vec{y}^*\vec{y}^{*\top}) \end{bmatrix}^{1/2}.$$

Since

$$H^- + H^{-\top} = \begin{bmatrix} -\nabla^2_{\vec{x}\vec{x}} f & 0 \\ 0 & \nabla^2_{\vec{y}\vec{y}} f \end{bmatrix}$$

by the convex-concave assumption on f it follows that the matrix above is negative semi-definite (see Remark 3.2.1) and so is $R + R^\top$. We conclude that the maximum eigenvalue of $R + R^\top$ is non-positive. Therefore any eigenvalue of R has real part non-positive and the same is true for J_{small}. □

Lemma 3.3.2. *If ϵ is a non-zero eigenvalue of J_{small} then, $Re(\epsilon) \leq 0$ and $|\epsilon| \downarrow 0$ as the stepsize $\eta \to 0$.*

We first can see that η which is the learning rate multiplies any eigenvalue and we may assume that whilst η is positive, it may be chosen to be sufficiently small and hence the magnitude of any eigenvalue $|\epsilon| \downarrow 0$.

Remark 3.3.1. *The equation $\epsilon = \frac{\lambda(\lambda-1)}{2\lambda-1}$ determines two complex roots for each fixed ϵ, say λ_1 and λ_2. The relation between $|\epsilon|$, $|\lambda_1|$ and $|\lambda_2|$ is illustrated in Figure 3.2, where the x-axis is taken to be $\propto \exp(1/|\epsilon|)$. Specifically we choose $\epsilon = -1/\log(x) + 1/\log(x)\sqrt{-1}$ that satisfies $|\epsilon| \downarrow 0$ as $x \to \infty$ (The x-axis of Figure 3.2 takes x from 3 to 103).*

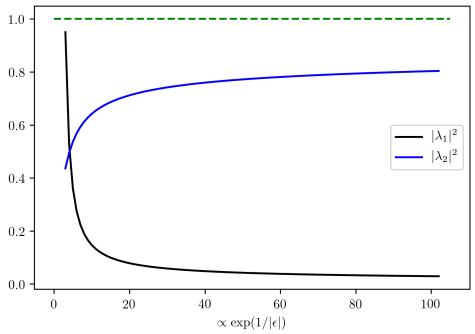


Figure 3.2: λ_1 and λ_2 less than 1 as $|\epsilon|$ is small.

Proof. Let $\lambda = x + \sqrt{-1}y$ and $\epsilon = a + \sqrt{-1}b$. The relation $\frac{\lambda(\lambda-1)}{2\lambda-1} = \epsilon$ gives two equations based on the equality of real and imaginary parts as follows,

$$x^2 - x - y^2 = 2ax - a - 2by \tag{3.10}$$

$$2xy - y = 2bx + 2ay - b. \tag{3.11}$$

Notice that the above equations can be transformed to the following forms:

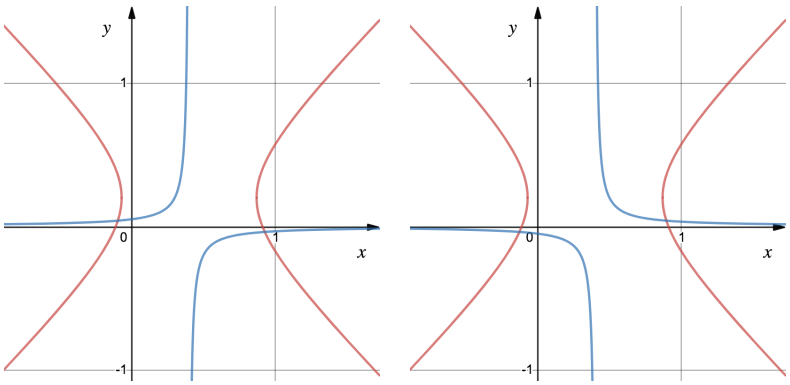
$$(x - \frac{2a+1}{2})^2 - (y-b)^2 = -a - b^2 + \frac{(2a+1)^2}{4} \tag{3.12}$$

$$(x - \frac{2a+1}{2})(y - b) = ab. \tag{3.13}$$

For each $\epsilon = a + \sqrt{-1}b$, there exist two pairs of points (x_1, y_1) and (x_2, y_2) that are the intersections of the above two hyperbola, illustrated in Figure 3.4. Recall the condition that $a < 0$. As $|\epsilon| \to 0$, the hyperbola can be obtained from the translation by $(\frac{2a+1}{2}, b)$ of the hyperbola

$$x^2 - y^2 = -a - b^2 + \frac{(2a+1)^2}{4}$$
$$xy = ab$$

where the translated symmetric center is close to $(\frac{1}{2}, 0)$ since (a, b) is close to $(0, 0)$. So the two intersections of the above hyperbola, (x_1, y_1) and (x_2, y_2), satisfy the property that $x_1^2 + y_1^2$ is small and $x_2 > \frac{1}{2}$ since the two intersections are on two sides of the axis $x = \frac{2a+1}{2}$, as showed in Figure 3.3. On the other hand, we have



$ab < 0$ $\qquad$ $ab > 0$

Figure 3.3: The intersections of the four branches of hyperbola are the two solutions of the equations (3.10) or (3.12). The intersections are on two sides of the line defined by $x = \frac{2a+1}{2}$, provided $|b|$ is small and $a < 0$. This occurs in the case either $ab > 0$ or $ab < 0$.

$$\frac{\lambda(\lambda - 1)}{2\lambda - 1} = \frac{(x + \sqrt{-1}y)(x - 1 + \sqrt{-1}y)}{2x - 1 + \sqrt{-1}2y} = \epsilon = a + \sqrt{-1}b$$

and then the condition $a < 0$ gives the inequality

$$\text{Re}(\epsilon) = \frac{(x^2 - x + y^2)(2x - 1)}{(2x - 1)^2 + 4y^2} < 0$$

that is equivalent to

$$x > \frac{1}{2} \quad \text{and} \quad x^2 - x + y^2 < 0$$

where only the case $x > \frac{1}{2}$ is considered since if the intersection whose x-component satisfying $x < \frac{1}{2}$ has the property that $x^2 + y^2$ is small and then less than 1, Figure 3.4. Thus to prove that $|\lambda| < 1$, it suffices to assume $x > \frac{1}{2}$. It is obvious that $x^2 - x + y^2 = (x - \frac{1}{2})^2 + y^2 - \frac{1}{4} < 0$ implies that $x^2 + y^2 < 1$. The proof completes. □

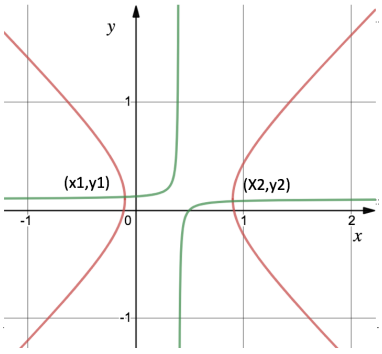
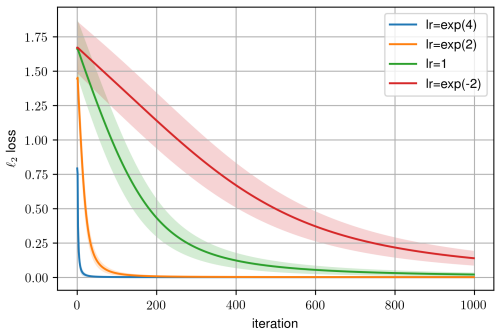
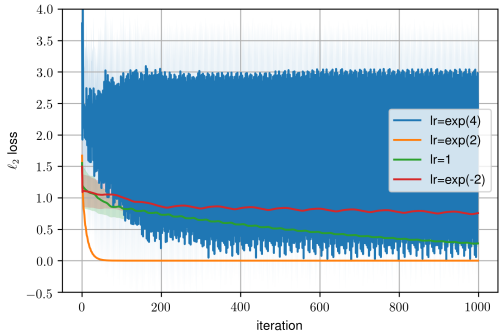


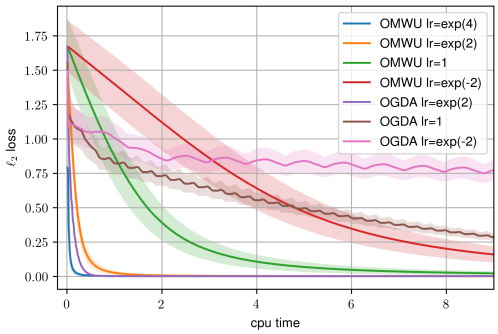
Figure 3.4: $a = -0.1$, $b = 0.1$



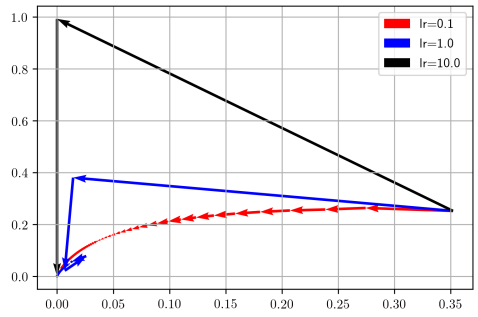
(a) OMWU



(b) OGDA

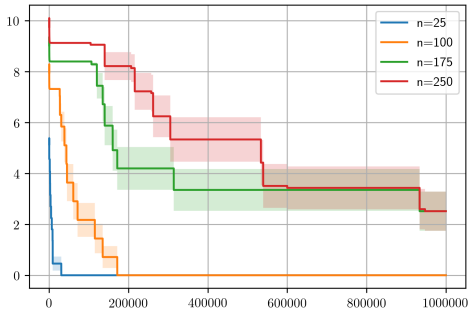


(c) Convergence time comparisons

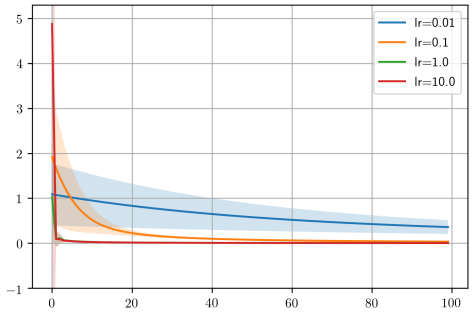


(d) OMWU trajectories with different learning rate

Figure 3.5: *Time comparisons of OMWU and projected OGDA vs different choices of learning rate.* For Figure (a)(b)(c), x-axis is iterations and y-axis is the l_1 error to the stationary point for Eqn. (3.14) with $n = 100$. We observe that OMWU (as in (a)) always converges while projected OGDA (as in (b)) will diverge for large learning rate. In figure (c) we remove the divergent case and compare the efficiency of the two algorithm measured in CPU time. In Figure (d) we visually present the trajectories for the min-max game of $\min_{\vec{x}\in\Delta_2}\max_{\vec{y}\in\Delta_2}\{x_1^2 - y_1^2 + 2x_1y_1\}$ with learning rate $0.1, 1.0$ and 10. Here x-axis is the value of x_1 and y-axis is the value of y_1 respectively. The equilibrium point the algorithm converges to is $\vec{x} = [0, 1], \vec{y} = [0, 1]$.



(a) KL divergence vs #iterations with different n



(b) KL divergence vs #iterations with different η

Figure 3.6: *KL divergence decreases with #iterations under different settings.* For both images, x-axis is the number of iterations, and y-axis is KL divergence. Figure (a) is OMWU on bilinear function Eqn.(3.14) with $n = \{25, 100, 175, 250\}$. Figure (b) is OMWU on the quadratic function $f(\vec{x}, \vec{y}) = x_1^2 - y_1^2 + 2x_1y_1$ with different learning rate η in $\{0.01, 0.1, 1.0, 10.0\}$. Shaded area indicates standard deviation from 10 runs with random initializations. OMWU with smaller learning rate tends to have higher variance.

3.4 Experiments

In this section, we conduct empirical studies to verify the theoretical results of our paper. We primarily target to understand two factors that influence the convergence speed of OMWU: the problem size and the learning rate. We also compare our algorithm with Optimistic Gradient Descent Ascent (OGDA) with projection, and demonstrate our superiority against it.

We start with a simple bilinear min-max game:

$$\min_{\vec{x}\in\Delta_n} \max_{\vec{y}\in\Delta_n} \vec{x}^\top A\vec{y}. \tag{3.14}$$

We first vary the value of n to study how the learning speed scales with the size of the problem. The learning rate is fixed at 1.0, and we run OMWU with $n \in \{25, 50, 75, \cdots, 250\}$ and matrix $A \in \mathbb{R}^{n\times n}$ is generated with i.i.d random Gaussian entries. We output the number of iterations for

OMWU to reach convergence, i.e., with l_1 error to the optimal solution to be less or equal to 10^{-5}. The results are averaged from 10 runs with different randomly initializations. As reported in Figure 3.1, generally a larger problem size requires more iterations to reach convergence. We also provide four specific cases of n to show the convergence in l_1 distance in Figure 3.1(b). The shaded area demonstrates the standard deviation from the 50 runs.

To understand how learning rate affects the speed of convergence, we conduct similar experiments on Eqn. (3.14) and plot the l_1 error with different step sizes in Figure 3.5(a)-(c). For this experiment the matrix size is fixed as $n = 100$. We also include a comparison with the Optimistic Gradient Descent Ascent[41]. Notice the original proposal was for unconstrained problems and we use projection in each step in order to constrain the iterates to stay inside the simplex. For the setting we considered, we observe a larger learning rate effectively speeds up our learning process, and our algorithm is relatively more stable to the choice of step-size. In comparison, OGDA is quite sensitive to the choice of step-size. As shown in Figure 3.5(b), a larger step-size makes the algorithm diverge, while a smaller step-size will make very little progress. Furthermore, we also choose to perform our algorithm over a convex-concave but not bilinear function $f(\vec{x}, \vec{y}) = x_1^2 - y_1^2 + 2x_1y_1$, where $\vec{x}, \vec{y} \in \Delta_2$ and x_1 and y_1 are the first coefficients of $\vec{x}$ and $\vec{y}$. With this low dimensional function, we could visually show the convergence procedure as in Figure 3.5(b), where each arrow indicates an OMWU step. This figure demonstrates that at least in this case, a larger step size usually makes sure a bigger progress towards the optimal solution.

Finally we show how the KL divergence $D_{KL}((\vec{x}^*, \vec{y}^*) \parallel (\vec{x}^t, \vec{y}^t))$ decreases under different circumstances. Figure 3.6 again considers the bilinear problem (Eqn.(3.14)) with multiple dimensions n and a simple convex-concave function $f(\vec{x}, \vec{y}) = x_1^2 - y_1^2 + 2x_1y_1$ with different learning rate. We note that in all circumstances we consider, we observe that OMWU is very stable, and

achieves global convergence invariant to the problem size, random initialization, and learning rate.

3.5 Conclusion

In this paper we analyze the last iterate behavior of a no-regret learning algorithm called Optimistic Multiplicative Weights Update for convex-concave landscapes. We prove that OMWU exhibits last iterate convergence in a neighborhood of the fixed point of OMWU algorithm, generalizing previous results that showed last iterate convergence for bilinear functions. The experiments explores how the problem size and the choice of learning rate affect the performance of our algorithm. We find that OMWU achieves global convergence and less sensitive to the choice of hyperparameter, compared to projected optimistic gradient descent ascent.

Chapter 4

Non-Convex-Concave Objective: On Learning Generative Models

Generative adversarial networks (GANs) are a widely used framework for learning generative models. Wasserstein GANs (WGANs), one of the most successful variants of GANs, require solving a minmax optimization problem to global optimality, but are in practice successfully trained using stochastic gradient descent-ascent. In this paper, we show that, when the generator is a one-layer network, stochastic gradient descent-ascent converges to a global solution with polynomial time and sample complexity.[1]

4.1 Introduction

Generative Adversarial Networks (GANs) [61] are a prominent framework for learning generative models of complex, real-world distributions given samples from these distributions. GANs and their variants have been successfully applied to numerous datasets and tasks, including image-to-image translation [69], image super-resolution [88], domain adaptation [157], probabilistic inference [50], compressed sensing [17, 92] and many more. These advances owe in part to the success of Wasserstein GANs (WGANs) [7, 65], leveraging the neural net induced integral probability metric to better measure the difference between a target and a generated distribution.

Along with the aforementioned empirical successes, there have been theoretical studies of the statistical properties of GANs—see e.g. [176, 9, 12, 13, 50] and their references. These works have shown that, with an appropriate design of the generator and discriminator, the global optimum of the WGAN objective identifies the target distribution with low sample complexity. However, these results cannot be algorithmically attained via practical GAN training algorithms.

[1]This work is based on the following arXiv paper:
Qi Lei, Jason D. Lee, Alexandros G. Dimakis, Constantinos Daskalakis. "SGD Learns One-Layer Networks in WGANs", arXiv preprint arXiv:1910.07030 [93]

On the algorithmic front, prior work has focused on the stability and convergence properties of gradient descent-ascent (GDA) and its variants in GAN training and more general min-max optimization problems; see e.g. [121, 67, 115, 114, 39, 40, 41, 59, 103, 118, 72, 105] and their references. These works have studied conditions under which GDA converges to a globally optimal solution in the convex-concave objective, or local stability in the non-convex non-concave setting. These results do not ensure convergence to a globally optimal generator, or in fact even convergence to a locally optimal generator.

Thus a natural question is whether:

Are GANs able to learn high-dimensional distributions in polynomial time and polynomial/parametric sample complexity, and thus bypass the curse of dimensionality?

The aforementioned prior works stop short of this goal due to a) the intractability of min-max optimization in the non-convex setting, and b) the curse of dimensionality in learning with Wasserstein distance in high dimensions [13].

A notable exception is [52] which shows that for WGANs with a linear generator and quadratic discriminator GDA succeeds in learning a Gaussian using polynomially many samples in the dimension.

In the same vein, we are the first to our knowledge to study the global convergence properties of stochastic GDA in the GAN setting, and establishing such guarantees for non-linear generators. In particular, we study the WGAN formulation for learning a single-layer generative model with some reasonable choices of activations including tanh, sigmoid and leaky ReLU.

Our contributions. For WGAN with a one-layer generator network using an activation from a large family of functions and a quadratic discriminator, we show that stochastic gradient

descent-ascent learns a target distribution using polynomial time and samples, under the assumption that the target distribution is realizable in the architecture of the generator. This is achieved by *simultaneously* satisfying the following two criterion:

1. Proving that stochastic gradient-descent attains a globally optimal generator in the metric induced by the discriminator,

2. Proving that appropriate design of the discriminator ensures a parametric $\mathcal{O}(\frac{1}{\sqrt{n}})$ statistical rate [176, 13] that matches the lower bound for learning one-layer generators as shown in [166].

4.2 Related Work

We briefly review relevant results in GAN training and learning generative models:

4.2.1 Optimization viewpoint

For standard GANs and WGANs with appropriate regularization, [121], [115] and [67] establish sufficient conditions to achieve local convergence and stability properties for GAN training. At the equilibrium point, if the Jacobian of the associated gradient vector field has only eigenvalues with negative real-part, GAN training is verified to converge locally for small enough learning rates. A follow-up paper by [114] shows the necessity of these conditions by identifying a counterexample that fails to converge locally for gradient descent based GAN optimization. The lack of global convergence prevents the analysis from yielding any guarantees for learning the real distribution.

The work of [52] described above has similar goals as our paper, namely understanding the convergence properties of basic dynamics in simple WGAN formulations. However, they only

consider linear generators, which restrict the WGAN model to learning a Gaussian. Our work goes a step further, considering WGANs whose generators are one-layer neural networks with a broad selection of activations. We show that with a proper gradient-based algorithm, we can still recover the ground truth parameters of the underlying distribution.

More broadly, WGANs typically result in nonconvex-nonconcave min-max optimization problems. In these problems, a global min-max solution may not exist, and there are various notions of local min-max solutions, namely local min-local max solutions [41], and local min solutions of the max objective [72], the latter being guaranteed to exist under mild conditions. In fact, [105] show that GDA is able to find stationary points of the max objective in nonconvex-concave objectives. Given that GDA may not even converge for convex-concave objectives, another line of work has studied variants of GDA that exhibit global convergence to the min-max solution [39, 40, 59, 103, 118], which is established for GDA variants that add negative momentum to the dynamics. While the convergence of GDA with negative momentum is shown in convex-concave settings, there is experimental evidence supporting that it improves GAN training [39, 59].

4.2.2 Statistical viewpoint

Several works have studied the issue of mode collapse. One might doubt the ability of GANs to actually learn the distribution vs just memorize the training data [9, 12, 50]. Some corresponding cures have been proposed. For instance, [176, 13] show for specific generators combined with appropriate parametric discriminator design, WGANs can attain parametric statistical rates, avoiding the exponential in dimension sample complexity [102, 13, 52].

Recent work of [166] provides an algorithm to learn the distribution of a single-layer ReLU generator network. While our conclusion appears similar, our focus is very different. Our

paper targets understanding when a WGAN formulation trained with stochastic GDA can learn in polynomial time and sample complexity. Their work instead relies on a specifically tailored algorithm for learning truncated normal distributions [38].

4.3 Preliminaries

Notation. We consider GAN formulations for learning a generator $G_A : \mathbb{R}^k \to \mathbb{R}^d$ of the form $\boldsymbol{z} \mapsto \boldsymbol{x} = \phi(A\boldsymbol{z})$, where A is a $d \times k$ parameter matrix and ϕ some activation function. We consider discriminators $D_{\boldsymbol{v}} : \mathbb{R}^d \to \mathbb{R}$ or $D_V : \mathbb{R}^d \to \mathbb{R}$ respectively when the discriminator functions are parametrized by either vectors or matrices. We assume latent variables $\boldsymbol{z}$ are sampled from the normal $\mathcal{N}(0, I_{k\times k})$, where $I_{k\times k}$ denotes the identity matrix of size k. The real/target distribution outputs samples $\boldsymbol{x} \sim \mathcal{D} = G_{A^*}(\mathcal{N}(0, I_{k_0\times k_0}))$, for some ground truth parameters A^*, where A^* is $d \times k_0$, and we take $k \geq k_0$ for enough expressivity, taking $k = d$ when k_0 is unknown.

The Wasserstain GAN under our choice of generator and discriminator is naturally formulated as:

$$\min_{A\in\mathbb{R}^{d\times k}} \max_{\boldsymbol{v}\in\mathbb{R}^d} f(A, \boldsymbol{v}),^{2}$$

for $f(A, \boldsymbol{v}) \equiv \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} D_{\boldsymbol{v}}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})} D_{\boldsymbol{v}}(G_A(\boldsymbol{z}))$.

We use $\boldsymbol{a}_i$ to denote the i-th row vector of A. We sometimes omit the 2 subscript, using $\|\boldsymbol{x}\|$ to denote the 2-norm of vector $\boldsymbol{x}$, and $\|X\|$ to denote the spectral norm of matrix X when there is no ambiguity. $\mathbb{S}^n \subset \mathbb{R}^{n\times n}$ represents all the symmetric matrices of dimension $n \times n$. We use $Df(X_0)[B]$ to denote the directional derivative of function f at point X_0 with direction B: $Df(X_0)[B] = \lim_{t\to 0} \frac{f(X_0+tB)-f(X_0)}{t}$.

[2] We will replace $\boldsymbol{v}$ by matrix parameters $V \in \mathbb{R}^{d\times d}$ when necessary.

4.3.1 Motivation and Discussion

To provably learn one-layer generators with nonlinear activations, the design of the discriminator must strike a delicate balance:

1. (Approximation.) The discriminator should be large enough to be able to distinguish the true distribution from incorrect generated ones. To be more specific, the max function $g(A) = \max_{\boldsymbol{x}} f(A, V)$ captures some distance from our learned generator to the target generators. This distance should only have global minima that correspond to the ground truth distribution.

2. (Generalizability.) The discriminator should be small enough so that it can be learned with few samples. In fact, our method guarantees an $\mathcal{O}(1/\sqrt{n})$ parametric rate that matches the lower bound established in [166].

3. (Stability.) The discriminator should be carefully designed so that simple local algorithms such as gradient descent ascent can find the global optimal point.

Further, min-max optimization with non-convexity in either side is intractable. In fact, gradient descent ascent does not even yield last iterate convergence for bilinear forms, and it requires more carefully designed algorithms like Optimistic Gradient Descent Ascent [41] and Extra-gradient methods [84]. In this paper we show a stronger hardness result. We show that for simple bilinear forms with ReLU activations, it is NP-hard to even find a stationary point.

Theorem 4.3.1. *Consider the min-max optimization on the following ReLU-bilinear form:*

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{y}} \left\{ f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} \phi(A_i \boldsymbol{x} + \boldsymbol{b}_i)^\top \boldsymbol{y} \right\},$$

where $\boldsymbol{x} \in \mathbb{R}^d$, $A_i \in \mathbb{R}^{\mathcal{O}(d)\times d}$ and ϕ is ReLU activation. As long as $n \geq 4$, the problem of checking whether f has any stationary point is NP-hard in d.

We defer the proof to the Appendix where we show 3SAT is reducible to the above problem. This theorem shows that in general, adding non-linearity (non-convexity) in min-max forms makes the problem intractable. However, we are able to show gradient descent ascent finds global minima for training one-layer generators with non-linearity. This will rely on carefully designed discriminators, regularization and specific structure that we considered.

Finally we note that understanding the process of learning one-layer generative model is important in practice as well. For instance, Progressive GAN [78] proposes the methodology to learn one-layer at a time, and grow both the generator and discriminator progressively during the learning process. Our analysis implies further theoretical support for this kind of progressive learning procedure.

4.4 Warm-up: Learning the Marginal Distributions

As a warm-up, we ask whether a simple linear discriminator is sufficient for the purposes of learning the marginal distributions of all coordinates of $\mathcal{D}$. Notice that in our setting, the i-th output of the generator is $\phi(x)$ where $x \sim \mathcal{N}(0, \|\boldsymbol{a}_i\|^2)$, and is thus solely determined by $\|\boldsymbol{a}_i\|_2$. With a linear discriminator $D_{\boldsymbol{v}}(\boldsymbol{x}) = \boldsymbol{v}^\top \boldsymbol{x}$, our minimax game becomes:

$$\min_{A\in\mathbb{R}^{d\times k}} \max_{\boldsymbol{v}\in\mathbb{R}^d} f_1(A, \boldsymbol{v}), \tag{4.1}$$

for $f_1(A, \boldsymbol{v}) \equiv \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\boldsymbol{v}^\top \boldsymbol{x}\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\boldsymbol{v}^\top \phi(A\boldsymbol{z})\right]$.

Notice that when the activation ϕ is an odd function, such as the tanh activation, the symmetric property of the Gaussian distribution ensures that $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{v}^\top \boldsymbol{x}] = 0$, hence the linear

discriminator in f_1 reveals no information about A^*. Therefore specifically for odd activations (or odd plus a constant activations), we instead use an adjusted rectified linear discriminator $D_{\boldsymbol{v}}(\boldsymbol{x}) \equiv \boldsymbol{v}^\top R(\boldsymbol{x} - C)$ to enforce some bias, where $C = \frac{1}{2}(\phi(x) + \phi(-x))$ for all x, and R denotes the ReLU activation. Formally, we slightly modify our loss function as:

$$\begin{aligned} \bar{f}_1(A, \boldsymbol{v}) \equiv & \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left[\boldsymbol{v}^\top R(\boldsymbol{x} - C)\right] \\ & - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0, I_{k\times k})} \left[\boldsymbol{v}^\top R(\phi(A\boldsymbol{z}) - C)\right]. \end{aligned} \tag{4.2}$$

We will show that we can learn each marginal of $\mathcal{D}$ if the activation function ϕ satisfies the following.

Assumption 4.4.1. *The activation function ϕ satisfies either one of the following:*

1. ϕ is an odd function plus constant, and ϕ is monotone increasing;

2. The even component of ϕ, i.e. $\frac{1}{2}(\phi(x) + \phi(-x))$, is positive and monotone increasing on $x \in [0, \infty)$.

Remark 4.4.1. *All common activation functions like (Leaky) ReLU, tanh or sigmoid function satisfy Assumption 4.4.1.*

Lemma 4.4.1. *Suppose $A^* \neq 0$. Consider f_1 with activation that satisfies Assumption 4.4.1.2 and $\bar{f}_1$ with activation that satisfies Assumption 4.4.1.1. The stationary points of such f_1 and $\bar{f}_1$ yield parameters A satisfying $\|\boldsymbol{a}_i\| = \|\boldsymbol{a}_i^*\|, \forall i \in [d]$.*

To bound the capacity of the discriminator, WGAN adds an Lipschitz constraint: $\|D_{\boldsymbol{v}}\| \leq 1$, or simply $\|\boldsymbol{v}\|_2 \leq 1$. To make the training process easier, we instead regularize the discriminator. For the regularized formulation we have:

Theorem 4.4.2. *In the same setting as Lemma 4.4.1, alternating gradient descent-ascent with proper learning rates on*

$$\min_{A}\max_{\boldsymbol{v}}\{f_1(A,\boldsymbol{v})-\|\boldsymbol{v}\|^2/2\},$$
$$\textit{or respectively}\qquad \min_{A}\max_{\boldsymbol{v}}\{\bar{f}_1(A,\boldsymbol{v})-\|\boldsymbol{v}\|^2/2\},$$

recovers A such that $\|\boldsymbol{a}_i\|=\|\boldsymbol{a}_i^*\|,\forall i\in[d]$.

All the proofs of the paper can be found in the appendix. We show that all local min-max points in the sense of [72] of the original problem are global min-max points and recover the correct norm of $\boldsymbol{a}_i^*,\forall i$. Notice for the source data distribution $\boldsymbol{x}=(x_1,x_2,\cdots x_d)\sim\mathcal{D}$ with activation ϕ, the marginal distribution of each x_i follows $\phi(\mathcal{N}(0,\|\boldsymbol{a}_i^*\|^2))$ and is determined by $\|\boldsymbol{a}_i^*\|$. Therefore we have learned the marginal distribution for each entry i. It remains to learn the joint distribution.

4.5 Learning the Joint Distribution

In the previous section, we utilize a (rectified) linear discriminator, such that each coordinate v_i interacts with the i-th random variable. With the (rectified) linear discriminator, WGAN learns the correct $\|\boldsymbol{a}_i\|$, for all i. However, since there's no interaction between different coordinates of the random vector, we do not expect to learn the joint distribution with a linear discriminator.

To proceed, a natural idea is to use a quadratic discriminator $D_V(\boldsymbol{x}):=\boldsymbol{x}^\top V\boldsymbol{x}=\langle\boldsymbol{x}\boldsymbol{x}^\top,V\rangle$ to enforce component interactions. Similar to the previous section, we study the regularized version:

$$\min_{A\in\mathbb{R}^{d\times k}}\max_{V\in\mathbb{R}^{d\times d}}\{f_2(A,V)-\frac{1}{2}\|V\|_F^2\},\tag{4.3}$$

where

$$f_2(A,V)$$

$$= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} D_V(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})} D_V(\phi(A\boldsymbol{z}))$$
$$= \left\langle \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left[\boldsymbol{x}\boldsymbol{x}^\top\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})} \left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right], V \right\rangle.$$

By adding a regularizer on V and explicitly maximizing over V:

$$\begin{aligned} g(A) &\equiv \max_V \left\{ f_2(A,V) - \frac{1}{2}\|V\|_F^2 \right\} \\ &= \frac{1}{2} \Big\| \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}} \left[\boldsymbol{x}\boldsymbol{x}^\top\right] \\ &\quad - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})} \left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right] \Big\|_F^2. \end{aligned}$$

In the next subsection, we first focus on analyzing the second-order stationary points of g, then we establish that gradient descent ascent converges to second-order stationary points of g .

4.5.1 Global Convergence for Optimizing the Generating Parameters

We first assume that both A and A^* have unit row vectors, and then extend to general case since we already know how to learn the row norms from Section 4.4. To explicitly compute $g(A)$, we rely on the property of Hermite polynomials. Since normalized Hermite polynomials $\{h_i\}_{i=0}^\infty$ forms an orthonomal basis in the functional space, we rewrite the activation function as $\phi(\boldsymbol{x}) = \sum_{i=0}^\infty \sigma_i h_i$, where σ_i is the i-th Hermite coefficient. We use the following claim:

Claim 4.5.1 ([58] Claim 4.2)**.** *Let ϕ be a function from $\mathbb{R}$ to $\mathbb{R}$ such that $\phi \in L^2(\mathbb{R}, e^{-x^2/2})$, and let its Hermite expansion be $\phi = \sum_{i=1}^\infty \sigma_i h_i$. Then, for any unit vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$, we have that*

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{N}(0,I_{d\times d})} \left[\phi(\boldsymbol{u}^\top\boldsymbol{x})\phi(\boldsymbol{v}^\top\boldsymbol{x})\right] = \sum_{i=0}^\infty \sigma_i^2 (\boldsymbol{u}^\top\boldsymbol{v})^i.$$

Therefore we could compute the value of f_2 explicitly using the Hermite polynomial expansion:

$$f_2(A,V) = \left\langle \sum_{i=0}^\infty \sigma_i^2 \left((A^*(A^*)^\top)^{\circ i} - (AA^\top)^{\circ i} \right), V \right\rangle.$$

Here $X^{\circ i}$ is the Hadamard power operation where $(X^{\circ i})_{jk} = (X_{jk})^i$. Therefore we have:

$$g(A) \quad = \quad \frac{1}{2}\left\|\sum_{i=0}^{\infty}\sigma_i^2\left((A^*(A^*)^\top)^{\circ i} - (AA^\top)^{\circ i}\right)\right\|_F^2$$

We reparametrize with $Z = AA^\top$ and define $\tilde{g}(Z) = g(A)$ with individual component functions $\tilde{g}_{jk}(z) \equiv \frac{1}{2}(\sum_{i=0}^{\infty}\sigma_i^2((z_{jk}^*)^i - z^i))^2$. Accordingly $z_{jk}^* = \langle \boldsymbol{a}_j^*, \boldsymbol{a}_k^*\rangle$ is the (j,k)-th component of the ground truth covariance matrix $A^*(A^*)^\top$.

Assumption 4.5.1. *The activation function ϕ is an odd function plus constant. In other words, its Hermite expansion $\phi = \sum_{i=0}^{\infty}\sigma_i h_i$ satisfies $\sigma_i = 0$ for even $i \geq 2$. Additionally we assume $\sigma_1 \neq 0$.*

Remark 4.5.1. *Common activations like tanh and sigmoid satisfy Assumption 4.5.1.*

Lemma 4.5.2. *For activations including leaky ReLU and functions satisfying Assumption 4.5.1, $\tilde{g}(Z)$ has a unique stationary point where $Z = A^*(A^*)^\top$.*

Notice $\tilde{g}(Z) = \sum_{jk}\tilde{g}_{jk}(z_{jk})$ is separable across z_{jk}, where each $\tilde{g}_{jk}$ is a polynomial scalar function. Lemma 4.5.2 comes from the fact that the only zero point for $\tilde{g}_{jk}'$ is $z_{jk} = z_{jk}^*$, for odd activation ϕ and leaky ReLU. Then we migrate this good property to the original problem we want to solve:

Problem 1. *We optimize over function g when $\|\boldsymbol{a}_i^*\| = 1, \forall i$:*

$$\min_A \left\{ g(A) \equiv \frac{1}{2}\left\|\sum_{i=0}^{\infty}\sigma_i^2\left((A^*(A^*)^\top)^{\circ i} - (AA^\top)^{\circ i}\right)\right\|_F^2 \right\}$$
$$\textit{s.t. } \boldsymbol{a}_i^\top \boldsymbol{a}_i = 1, \forall i.$$

Existing work [75] connects $\tilde{g}(Z)$ to the optimization over factorized version for $g(A)$ ($g(A) \equiv \tilde{g}(AA^\top)$). Specifically, when $k = d$, all second-order stationary points for $g(A)$ are first-order stationary points for $\tilde{g}(Z)$. Though $\tilde{g}$ is not convex, we are able to show that its first-order stationary points are global optima when the generator is sufficiently expressive, i.e., $k \geq k_0$. In reality we won't know the latent dimension k_0, therefore we just choose $k = d$ for simplicity. We get the following conclusion:

Theorem 4.5.3. *For activations including leaky ReLU and functions satisfying Assumption 4.5.1, when $k = d$, all second-order KKT points for problem 1 are global minima. Therefore alternating projected gradient descent-ascent on Eqn. (4.3) converges to A such that $AA^\top = A^*(A^*)^\top$.*

The extension for non-unit vectors is straightforward, and we defer the analysis to the Appendix.

This main theorem demonstrates the success of gradient descent ascent on learning the ground truth generator. This result is achieved by analyzing two factors. One is the geometric property of our loss function, i.e., all second-order KKT points are global minima. Second, all global minima satisfy $AA^\top = A^*(A^*)^\top$, and for the problem we considered, i.e., one-layer generators, retrieving parameter $AA^\top$ is sufficient in learning the whole generating distribution.

4.6 Finite Sample Analysis

In the previous section, we demonstrate the success of using gradient descent ascent on the population risk. This leaves us the question on how many samples do we need to achieve small error. In this section, we analyze Algorithm 3, i.e., gradient descent ascent on the following empirical

Algorithm 3 Online stochastic gradient descent ascent on WGAN

1: **Input:** n training samples: $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots \boldsymbol{x}_n$, where each $\boldsymbol{x}_i \sim \phi(A^*\boldsymbol{z}), \boldsymbol{z} \sim \mathcal{N}(0, I_{k\times k})$, learning rate for generating parameters η, number of iterations T.
2: Random initialize generating matrix $A^{(0)}$.
3: **for** $t = 1, 2, \cdots, T$ **do**
4: Generate m latent variables $\boldsymbol{z}_1^{(t)}, \boldsymbol{z}_2^{(t)}, \cdots, \boldsymbol{z}_m^{(t)} \sim \mathcal{N}(0, I_{k\times k})$ for the generator. The empirical function becomes

$$\tilde{f}_{m,n}^{(t)}(A, V) = \left\langle \frac{1}{m}\sum_{i=1}^{m} \phi(A\boldsymbol{z}_i^{(t)})\phi(A\boldsymbol{z}_i^{(t)})^\top - \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^\top, V \right\rangle - \frac{1}{2}\|V\|^2$$

5: Gradient ascent on V with optimal step-size $\eta_V = 1$:

$$V^{(t)} \leftarrow V^{(t)} - \eta_V \nabla_V \tilde{f}_{m,n}^{(t)}(A^{(t-1)}, V^{(t-1)}).$$

6: Sample noise $\boldsymbol{e}$ uniformly from unit sphere
7: Projected Gradient Descent on A, with constraints $C = \{A|(AA^\top)_{ii} = (A^*A^{*\top})_{ii}\}$:

$$A^{(t)} \leftarrow \text{Proj}_C(A^{(t-1)} - \eta(\nabla_A \tilde{f}_{m,n}^{(t)}(A^{(t-1)}, V^{(t)}) + \boldsymbol{e})).$$

8: **end for**
9: **Output:** $A^{(T)}(A^{(T)})^\top$

loss:

$$\begin{aligned}
&\tilde{f}_{m,n}^{(t)}(A, V) \\
=&\left\langle \frac{1}{m}\sum_{i=1}^{m} \phi(A\boldsymbol{z}_i^{(t)})\phi(A\boldsymbol{z}_i^{(t)})^\top - \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^\top, V \right\rangle \\
&- \frac{1}{2}\|V\|^2.
\end{aligned}$$

Notice in each iteration, gradient ascent with step-size 1 finds the optimal solution for V. By Danskin's theorem [36], our min-max optimization is essentially gradient descent over $\tilde{g}_{m,n}^{(t)}(A) \equiv \max_V \tilde{f}_{m,n}^{(t)}(A, V) = \frac{1}{2}\|\frac{1}{m}\sum_{i=1}^{m} \phi(A\boldsymbol{z}_i^{(t)})\phi(A\boldsymbol{z}_i^{(t)})^\top - \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^\top\|_F^2$ with a batch of samples

$\{\boldsymbol{z}_i^{(t)}\}$, i.e., stochastic gradient descent for $f_n(A) \equiv \mathbb{E}_{\boldsymbol{z}_i \sim \mathcal{N}(0, I_{k\times k}), \forall i \in [m]}[\tilde{g}_{m,n}(A)]$.

Therefore to bound the difference between $f_n(A)$ and the population risk $g(A)$, we analyze the sample complexity required on the observation side ($\boldsymbol{x}_i \sim \mathcal{D}, i \in [n]$) and the mini-batch size required on the learning part ($\phi(A\boldsymbol{z}_j), \boldsymbol{z}_j \sim \mathcal{N}(0, I_{k\times k}), j \in [m]$). We will show that with large enough n, m, the algorithm specified in Algorithm 3 that optimizes over the empirical risk will yield the ground truth covariance matrix with high probability.

Our proof sketch is roughly as follows:

1. With high probability, projected stochastic gradient descent finds a second order stationary point $\hat{A}$ of $f_n(\cdot)$ as shown in Theorem 31 of [57].

2. For sufficiently large m, our empirical objective, though a biased estimator of the population risk $g(\cdot)$, achieves good ϵ-approximation to the population risk on both the gradient and Hessian (Lemmas 4.6.3&4.6.4). Therefore $\hat{A}$ is also an $\mathcal{O}(\epsilon)$-approximate second order stationary point (SOSP) for the population risk $g(A)$.

3. We show that any ϵ-SOSP $\hat{A}$ for $g(A)$ yields an $\mathcal{O}(\epsilon)$-first order stationary point (FOSP) $\hat{Z} \equiv \hat{A}\hat{A}^\top$ for the semi-definite programming on $\tilde{g}(Z)$ (Lemma 4.6.7).

4. We show that any $\mathcal{O}(\epsilon)$-FOSP of function $\tilde{g}(Z)$ induces at most $\mathcal{O}(\epsilon)$ absolute error compared to the ground truth covariance matrix $Z^* = A^*(A^*)^\top$ (Lemma 4.6.8).

4.6.1 Observation Sample Complexity

For simplicity, we assume the activation and its gradient satisfy Lipschitz continuous, and let the Lipschitz constants be 1 w.l.o.g.:

Assumption 4.6.1. *Assume the activation is* 1*-Lipschitz and* 1*-smooth.*

To estimate observation sample complexity, we will bound the gradient and Hessian for the population risk and empirical risk on the observation samples:

$$\begin{aligned}
&g(A)\\
\equiv&\frac{1}{2}\left\|\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\boldsymbol{x}\boldsymbol{x}^\top\right]-\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\|_F^2,\\
&g_n(A)\\
\equiv&\frac{1}{2}\left\|\frac{1}{n}\sum_{i=1}^n\boldsymbol{x}_i\boldsymbol{x}_i^\top-\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\|_F^2.
\end{aligned}$$

We calculate the gradient estimation error due to finite samples.

Claim 4.6.1.

$$\begin{aligned}
&\nabla g(A)-\nabla g_n(A)\\
=\;&2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi'(A\boldsymbol{z}))(X-X_n)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right],
\end{aligned}$$

where $X=\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}\boldsymbol{x}^\top]$, *and* $X_n=\frac{1}{n}\sum_{i=1}^n\boldsymbol{x}_i\boldsymbol{x}_i^\top$. *The directional derivative with arbitrary direction* B *is:*

$$\begin{aligned}
&D\nabla g(A)[B]-D\nabla g_n(A)[B]\\
=&2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi'(A\boldsymbol{z}))(X_n-X)\phi'(A\boldsymbol{z})\circ(B\boldsymbol{z})\boldsymbol{z}^\top\right]\\
&+2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi''(A\boldsymbol{z})\circ(B\boldsymbol{z}))(X_n-X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right]
\end{aligned}$$

Lemma 4.6.2. *Suppose the activation satisfies Assumption 4.6.1. We get*

$$\Pr[\|X-X_n\|\le\epsilon\|X\|]\ge1-\delta,$$

for $n\ge\tilde{\Theta}(d/\epsilon^2\log^2(1/\delta))$[3].

[3]We will use $\tilde{\Theta}$ throughout the paper to hide log factors of d for simplicity.

Bounding the relative difference between sample and population covariance matrices is essential for us to bound the estimation error in both gradient and its directional derivative. We can show the following relative error:

Lemma 4.6.3. *Suppose the activation satisfies Assumption 4.5.1&4.6.1. With samples* $n \geq \tilde{\Theta}(d/\epsilon^2 \log^2(1/\delta))$*, we get:*

$$\|\nabla g(A) - \nabla g_n(A)\|_2 \leq \mathcal{O}(\epsilon d \|A\|_2),$$

with probability $1 - \delta$*. Meanwhile,*

$$\|D\nabla g(A)[B] - D\nabla g_n(A)[B]\|_2 \leq \mathcal{O}(\epsilon d^{3/2} \|A\|_2 \|B\|_2),$$

with probability $1 - \delta$*.*

4.6.2 Bounding Mini-batch Size

Normally for empirical risk for supervised learning, the mini-batch size can be arbitrarily small since the estimator of the gradient is unbiased. However in the WGAN setting, notice for each iteration, we randomly sample a batch of random variables $\{\boldsymbol{z}_i\}_{i\in[m]}$, and obtain a gradient of

$$\tilde{g}_{m,n}(A) \equiv \frac{1}{2} \left\| \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top - \frac{1}{m} \sum_{j=1}^{m} \phi(A\boldsymbol{z}_j)\phi(A\boldsymbol{z}_j)^\top \right\|_F^2,$$

in Algorithm 3. However, the finite sum is inside the Frobenius norm and the gradient on each mini-batch may no longer be an unbiased estimator for our target

$$g_n(A) = \frac{1}{2} \left\| \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top - \mathbb{E}_{\boldsymbol{z}} \left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top \right] \right\|_F^2.$$

In other words, we conduct stochastic gradient descent over the function $f(A) \equiv \mathbb{E}_{\boldsymbol{z}} \tilde{g}_{m,n}(A)$. Therefore we just need to analyze the gradient error between this $f(A)$ and $g_n(A)$ (i.e. $\tilde{g}_{m,n}$ is almost an unbiased estimator of g_n). Finally with the concentration bound derived in last section, we get the error bound between $f(A)$ and $g(A)$.

Lemma 4.6.4. *The empirical risk $\tilde{g}_{m,n}$ is almost an unbiased estimator of g_n. Specifically, the expected function $f(A) = \mathbb{E}_{\boldsymbol{z}_i \sim \mathcal{N}(0, I_{k\times k}), i\in[m]}[\tilde{g}_{m,n}]$ satisfies:*

$$\|\nabla f(A) - \nabla g_n(A)\| \leq \mathcal{O}(\frac{1}{m}\|A\|^3 d^2).$$

For arbitrary direction matrix B,

$$\|D\nabla f(A)[B] - D\nabla g_n(A)[B]\| \leq \mathcal{O}(\frac{1}{m}\|B\|\|A\|^3 d^{5/2}).$$

In summary, we conduct concentration bound over the observation samples and mini-batch sizes, and show the gradient of $f(A)$ that Algorithm 3 is optimizing over has close gradient and Hessian with the population risk $g(A)$. Therefore a second-order stationary point (SOSP) for $f(A)$ (that our algorithm is guaranteed to achieve) is also an ϵ approximated SOSP for $g(A)$. Next we show such a point also yield an ϵ approximated first-order stationary point of the reparametrized function $\tilde{g}(Z) \equiv g(A), \forall Z = AA^\top$.

4.6.3 Relation on Approximate Optimality

In this section, we establish the relationship between $\tilde{g}$ and g. We present the general form of our target Problem 1:

$$\min_{A\in\mathbb{R}^{d\times k}} g(A) \equiv \tilde{g}(AA^\top) \tag{4.4}$$

$$\text{s.t.}\mathrm{Tr}(A^\top X_i A) = y_i, X_i \in \mathbb{S}, y_i \in \mathbb{R}, i = 1, \cdots, n.$$

Similar to the previous section, the stationary property might not be obvious on the original problem. Instead, we could look at the re-parametrized version as:

$$\begin{aligned} \min_{Z\in\mathbb{S}} \quad & \tilde{g}(Z) \\ \text{s.t.} \quad & \mathrm{Tr}(X_i Z) = y_i, X_i \in \mathbb{S}, y_i \in \mathbb{R}, i = 1, \cdots, n, \\ & Z \succeq 0, \end{aligned} \tag{4.5}$$

Definition 4.6.5. *A matrix $A \in \mathbb{R}^{d\times k}$ is called an ϵ-approximate second-order stationary point (ϵ-SOSP) of Eqn. (4.4) if there exists a vector λ such that:*

$$\begin{cases} \mathit{Tr}(A^\top X_i A) = y_i, i \in [n] \\ \|(\nabla_Z \tilde{g}(AA^\top) - \sum_{i=1}^n \lambda_i X_i)\tilde{\boldsymbol{a}}_j\| \le \epsilon\|\tilde{\boldsymbol{a}}_j\|, \\ \qquad\qquad (\{\tilde{\boldsymbol{a}}_j\}_j \textit{ span the column space of } A) \\ \mathit{Tr}(B^\top D\nabla_A \mathcal{L}(A,\lambda)[B]) \ge -\epsilon\|B\|^2, \\ \qquad\qquad \forall B \textit{ s.t. } \mathit{Tr}(B^\top X_i A) = 0 \end{cases}$$

Here $\mathcal{L}(A,\lambda)$ is the Lagrangian form $\tilde{g}(AA^\top) - \sum_{i=1}^n \lambda_i(\mathit{Tr}(A^\top X_i A) - y_i)$.

Specifically, when $\epsilon = 0$ the above definition is exactly the second-order KKT condition for optimizing (4.4). Next we present the approximate first-order KKT condition for (4.5):

Definition 4.6.6. *A symmetric matrix $Z \in \mathbb{S}^n$ is an ϵ-approximate first order stationary point of function (4.5) (ϵ-FOSP) if and only if there exist a vector $\sigma \in \mathbb{R}^m$ and a symmetric matrix $S \in \mathbb{S}$ such that the following holds:*

$$\begin{cases} \mathit{Tr}(X_i Z) = y_i, i \in [n] \\ Z \succeq 0, \\ S \succeq -\epsilon I, \\ \|S\tilde{\boldsymbol{a}}_j\| \le \epsilon\|\tilde{\boldsymbol{a}}_j\|, \\ \qquad\qquad (\{\tilde{\boldsymbol{a}}_j\}_j \textit{ span the column space of } Z) \\ S = \nabla_Z \tilde{g}(Z) - \sum_{i=1}^n \sigma_i X_i. \end{cases}$$

Lemma 4.6.7. *Let latent dimension $k = d$. For an ϵ-SOSP of function (4.4) with A and λ, it infers an ϵ-FOSP of function (4.5) with Z, σ and S that satisfies: $Z = AA^\top, \sigma = \lambda$ and $S = \nabla_Z \tilde{g}(AA^\top) - \sum_i \lambda_i X_i$.*

Now it remains to show an ϵ-FOSP of $\tilde{g}(Z)$ indeed yields a good approximation for the ground truth parameter matrix.

Lemma 4.6.8. *If Z is an ϵ-FOSP of function (4.5), then $\|Z - Z^*\|_F \leq \mathcal{O}(\epsilon)$. Here $Z^* = A^*(A^*)^\top$ is the optimal solution for function (4.5).*

Together with the previous arguments, we finally achieve our main theorem on connecting the recovery guarantees with the sample complexity and batch size[4]:

Theorem 4.6.9. *For arbitrary $\delta < 1, \epsilon$, given small enough learning rate $\eta < 1/\text{poly}(d, 1/\epsilon, \log(1/\delta))$, let sample size $n \geq \tilde{\Theta}(d^5/\epsilon^2 \log^2(1/\delta))$, batch size $m \geq \mathcal{O}(d^5/\epsilon)$, for large enough T=poly$(1/\eta, 1/\epsilon, d, \log(1/\delta))$, the output of Algorithm 3 satisfies*

$$\|A^{(T)}(A^{(T)})^\top - Z^*\|_F \leq \mathcal{O}(\epsilon),$$

with probability $1 - \delta$, under Assumptions 4.5.1 & 4.6.1 and $k = d$.

Therefore we have shown that with finite samples of poly$(d, 1/\epsilon)$, we are able to learn the generating distribution with error measured in the parameter space, using stochastic gradient descent ascent. This echos the empirical success of training WGAN. Meanwhile, notice our error bound matches the lower bound on dependence of $1/\epsilon$, as suggested in [166].

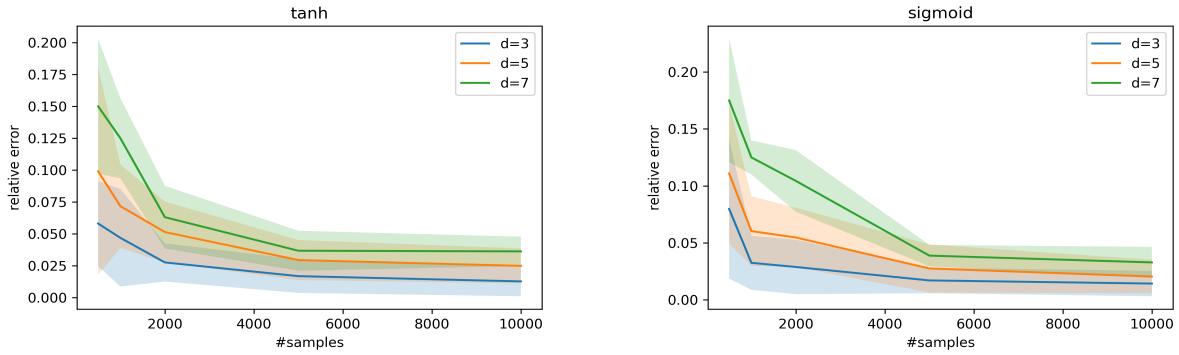
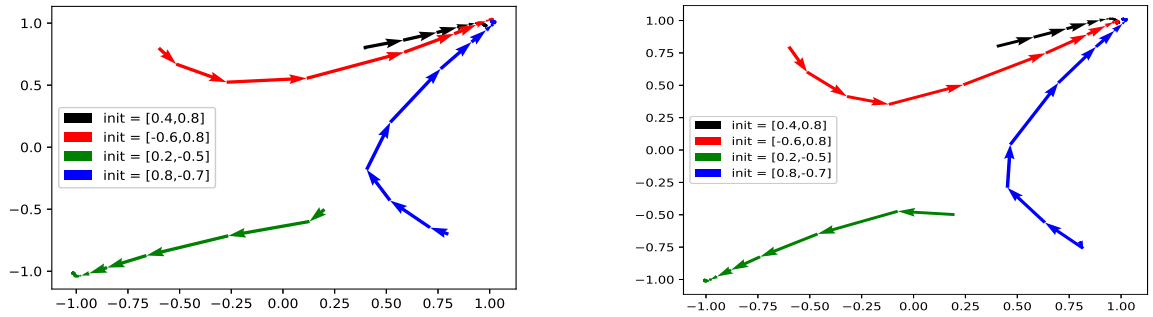


Figure 4.1: Recovery error ($\|AA^\top - Z^*\|_F$) with different observed sample sizes n and output dimension d.



(a) leaky ReLU activation ($\alpha = 0.2$)

(b) tanh activation

Figure 4.2: Comparisons of different performance with leakyReLU and tanh activations. Same color starts from the same starting point. For both cases, parameters always converge to true covariance matrix. Each arrow indicates the progress of 500 iteration steps.

4.7 Experiments

In this section, we provide simple experimental results to validate the performance of stochastic gradient descent ascent and provide experimental support for our theory.

[4]The exact error bound comes from the fact that when diagonal terms of $AA^\top$ are fixed, $\|A\|_2 = \mathcal{O}(\sqrt{d})$.

We focus on Algorithm 3 that targets to recover the parameter matrix. We conduct a thorough empirical studies on three joint factors that might affect the performance: the number of observed samples m (we set $n = m$ as in general GAN training algorithms), the different choices of activation function ϕ, and the output dimension d.

In Figure 4.1 we plot the relative error for parameter estimation decrease over the increasing sample complexity. We fix the hidden dimension $k = 2$, and vary the output dimension over $\{3, 5, 7\}$ and sample complexity over $\{500, 1000, 2000, 5000, 10000\}$. Reported values are averaged from 20 runs and we show the standard deviation with the corresponding colored shadow. Clearly the recovery error decreases with higher sample complexity and smaller output dimension. From the experimental results, we can see that our algorithm always achieves global convergence to the ground truth generators from any random initialization point.

To visually demonstrate the learning process, we also include a simple comparison for different ϕ: i.e. leaky ReLU and tanh activations, when $k = 1$ and $d = 2$. We set the ground truth covariance matrix to be $[1, 1; 1, 1]$, and therefore a valid result should be $[1, 1]$ or $[-1, -1]$. From Figure 4.2 we could see that for both leaky ReLU and tanh, the stochastic gradient descent ascent performs similarly with exact recovery of the ground truth parameters.

4.8 Conclusion

We analyze the convergence of stochastic gradient descent ascent for Wasserstein GAN on learning a single layer generator network. We show that stochastic gradient descent ascent algorithm attains the global min-max point, and provably recovers the parameters of the network with ϵ absolute error measured in Frobenius norm, from $\tilde{\Theta}(d^5/\epsilon^2)$ i.i.d samples.

Appendices

Appendix A

Appendix for Primal-Dual Generalized Block Frank-Wolfe

A.1 Omitted Proofs for Primal Dual Generalized Block Frank-Wolfe

A.1.1 Notation and simple facts

Recall primal, dual and Lagrangian forms:

$$\begin{aligned} P(\boldsymbol{x}) &\triangleq f^*(A\boldsymbol{x}) + g(\boldsymbol{x}) \\ \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) &\triangleq g(\boldsymbol{x}) + \boldsymbol{y}^\top A\boldsymbol{x} - f(\boldsymbol{y}) \\ D(\boldsymbol{y}) &\triangleq \min_{\boldsymbol{x}\in C} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{L}(\bar{\boldsymbol{x}}(\boldsymbol{y}), \boldsymbol{y}) \end{aligned}$$

Similar to the definitions in [97], we introduce the primal gap defined as $\Delta_p^{(t)} \triangleq \mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - D(\boldsymbol{y}^{(t)})$, and dual gap $\Delta_d^{(t)} \triangleq D^* - D(\boldsymbol{y}^{(t)})$. Recall the assumptions:

- f is $1/\beta$-strongly convex and is $1/\alpha$-smooth on a convex set and infinity otherwise.
- $R = \max_i \|\boldsymbol{a}_i\|_2^2, \forall i \in [n]$.
- g is μ-strongly convex and L-smooth.

For simplicity we first assume $\alpha \geq \frac{1}{2}\beta$ and then generalize the result.

Claim A.1.1.

- *Since $D(\boldsymbol{y}) = \min_{\boldsymbol{x}\in C}\{g(\boldsymbol{x}) + \boldsymbol{y}^\top A\boldsymbol{x}\} - f(\boldsymbol{y})$, $-D(\boldsymbol{y})$ is $\frac{1}{\beta}$-strongly convex.*
- *Based on our update rule, $\exists \boldsymbol{g} \in \partial_{\boldsymbol{y}} f(\boldsymbol{y}^{(t)})$, such that*

$$\boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)} = \delta(A_{I^{(t)},:}\boldsymbol{x}^{(t)} - \boldsymbol{g}_{I^{(t)}}). \tag{A.1}$$

And our update rule ensures that $I^{(t)}$ consists of indices $i \in [n]$ that maximizes $|\boldsymbol{a}_i^\top \boldsymbol{x}^{(t)} - g_i|$.

A.1.2 Primal Progress

Lemma A.1.2. *(Primal Progress)*

$$\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)}) \leq (1 - \frac{\eta}{2}) \left(\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})\right)$$

Or equivalently,

$$(1 - \frac{\eta}{2})(\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})) \leq -\frac{\eta}{2} \left(\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})\right) \equiv -\frac{\eta}{2}\Delta_p^{(t)}$$

Proof. Simply replace h_t as $\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - D(\boldsymbol{y}^{(t)})$ and h_{t+1} as $\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - D(\boldsymbol{y}^{(t)})$ in Inequality (2.7). We could conclude that $h_{t+1} \leq (1-\eta+\eta^2\frac{L}{\mu})h_t$. Therefore when $\eta \leq \frac{\mu}{2L}$, $h_{t+1} \leq (1-\frac{\eta}{2})h_t$ and the first part of Lemma A.1.2 is true. Some simple rearrangement suffices the second part of the lemma. □

A.1.3 Primal Dual Progress

In order to get a clue on how to analyze the dual progress, we first look at how the primal and dual evolve through iterations.

For an index set I and a vector $\boldsymbol{y} \in \mathbb{R}^n$, denote $\boldsymbol{y}_I = \sum_{i \in I} y_i \boldsymbol{e}_i \in \mathbb{R}^k$ as the subarray of $\boldsymbol{y}$ indexed by I, with $|I| = k$. Recall Algorithm 1 selects the coordinates to update in the dual variable as $I^{(t)}$.

Lemma A.1.3. *(Primal-Dual Progress).*

$$\begin{aligned} & \Delta_d^{(t)} - \Delta_d^{(t-1)} + \Delta_p^{(t)} - \Delta_p^{(t-1)} \\ \leq\ & \mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \frac{1}{2\delta}\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\ & +2\delta Rk\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2. \end{aligned}$$

Proof. Notice we have claimed that $-D(\boldsymbol{y})$ is $\frac{1}{\beta}$-strongly convex and for all $\boldsymbol{g} \in \partial_{\boldsymbol{y}} f(\boldsymbol{y}^{(t)})$,

$$\begin{aligned}
& \Delta_d^{(t)} - \Delta_d^{(t-1)} = \big(- D(\boldsymbol{y}^{(t)})\big) - \big(- D(\boldsymbol{y}^{(t-1)})\big) \\
\leq\ & \langle -\nabla_{\boldsymbol{y}} \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)}), \boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)} \rangle - \frac{1}{2\beta} \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
=\ & -\langle A_{I^{(t)},:} \bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}_{I^{(t)}}, \boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)} \rangle - \frac{1}{2\beta} \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2
\end{aligned} \tag{A.2}$$

Meanwhile since $-\mathcal{L}(\boldsymbol{x}, \boldsymbol{y})$ is $\frac{1}{\alpha}$-smooth over its feasible set,

$$\begin{aligned}
& \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t-1)}) \\
=\ & -\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t-1)}) - (-\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})) \\
\leq\ & (A_{I^{(t)},:} \boldsymbol{x}^{(t)} - \boldsymbol{g}_{I^{(t)}})^\top (\boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)}) + \frac{1}{2\alpha} \|\boldsymbol{y}_{I^{(t)}}^{(t-1)} - \boldsymbol{y}_{I^{(t)}}^{(t)}\|^2 \\
=\ & (\frac{1}{\delta} + \frac{1}{2\alpha}) \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2.
\end{aligned} \tag{A.3}$$

Also, with the update rule of dual variables, we could make use of Eqn. (A.1) and re-write Eqn. (A.2) as:

$$\begin{aligned}
& \Delta_d^{(t)} - \Delta_d^{(t-1)} \\
\leq\ & -\langle A_{I^{(t)},:} \bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}_{I^{(t)}}, \boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)} \rangle - \frac{1}{\delta} \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
& + (\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)})^\top (A_{I^{(t)},:} \boldsymbol{x}^{(t)} - \boldsymbol{g}_{I^{(t)}}) - \frac{1}{2\beta} \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
=\ & -\langle A_{I^{(t)},:} (\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}), \boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)} \rangle - (\frac{1}{\delta} + \frac{1}{2\beta}) \|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2
\end{aligned} \tag{A.4}$$

Together we get:

$$\begin{aligned}
& \Delta_d^{(t)} - \Delta_d^{(t-1)} + \Delta_p^{(t)} - \Delta_p^{(t-1)} \\
=& \mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) + \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t-1)})
\end{aligned}$$

$$
\begin{aligned}
&+ 2(\Delta_d^{(t)} - \Delta_d^{(t-1)}) \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) + (\frac{1}{\delta} + \frac{1}{2\alpha})\|\boldsymbol{y}_{I^{(t)}}^{(t-1)} - \boldsymbol{y}_{I^{(t)}}^{(t)}\|^2 + 2(\Delta_d^{(t)} - \Delta_d^{(t-1)}) \\
& \qquad \text{(from Eqn. (A.3))} \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) + (\frac{1}{\delta} + \frac{1}{2\alpha})\|\boldsymbol{y}_{I^{(t)}}^{(t-1)} - \boldsymbol{y}_{I^{(t)}}^{(t)}\|^2 \\
&- 2\langle A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}), \boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)}\rangle - 2(\frac{1}{\delta} + \frac{1}{2\beta})\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
& \qquad \text{(from Eqn. (A.4))} \\
= &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - 2\langle A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}), \boldsymbol{y}_{I^{(t)}}^{(t)} - \boldsymbol{y}_{I^{(t)}}^{(t-1)}\rangle \\
&- (\frac{1}{\delta} + \frac{1}{\beta} - \frac{1}{2\alpha})\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) + 2\delta\|A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)})\|^2 \\
&- (\frac{1}{\delta} - \frac{1}{2\delta})\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \qquad \text{(since } 2ab \leq \gamma a^2 + 1/\gamma b^2\text{)} \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \frac{1}{2\delta}\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
&+ 2\delta Rk\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2
\end{aligned}
$$

□

Therefore we will connect the progress induced by $-\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|$ and dual gap $\Delta_d^{(t)}$ next.

A.1.4 Dual progress

Claim A.1.4. *An α-strongly convex function f satisfies:*

$$f(\boldsymbol{x}) - f^* \leq \frac{1}{2\alpha}\|\nabla f(\boldsymbol{x})\|_2^2$$

This simply due to $f(\boldsymbol{x}) - f^* \leq \langle \nabla f(\boldsymbol{x}), \boldsymbol{x} - \bar{\boldsymbol{x}}\rangle - \frac{\alpha}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_2^2 \leq \frac{1}{2\alpha}\|\nabla f(\boldsymbol{x})\|^2 + \frac{\alpha}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 - \frac{\alpha}{2}\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|^2 = \frac{1}{2\alpha}\|\nabla f(\boldsymbol{x})\|^2$.

Since $-D$ is $\frac{1}{\beta}$-strongly convex, we get

$$\begin{aligned}\Delta_d^{(t)} = D^* - D(\boldsymbol{y}^{(t)}) \leq & \frac{\beta}{2}\|\nabla D(\boldsymbol{y}^{(t)})\|_2^2 \\ = & \frac{\beta}{2}\|A\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}\|_2^2 \\ \leq & \frac{n\beta}{2k}\|A_{\bar{I},:}\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}_{\bar{I}}\|_2^2, \end{aligned} \tag{A.5}$$

where $\bar{I}$ is a set of size k that maximizes the values of $A_i^\top \bar{\boldsymbol{x}}^{(t)} - g_i$.

Lemma A.1.5 (Dual Progress)**.**

$$-\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \leq -\frac{k\delta}{\beta}\Delta_d^{(t)} + k\delta R\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2$$

Proof of Lemma A.1.5. Define $\Delta = A(\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)})$. Since

$$\begin{aligned} & - \|A_{I^{(t)}}^\top \boldsymbol{x}^{(t)} - \boldsymbol{g}_{I^{(t)}}\|^2 \\ \leq & - \|A_{\bar{I}}^\top \boldsymbol{x}^{(t)} - \boldsymbol{g}_{\bar{I}}\|^2 && \text{(choice of } I^{(t)}\text{)} \\ = & - \|A_{\bar{I}}^\top \bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}_{\bar{I}} - \Delta_{\bar{I}}\|^2 \\ \leq & - \frac{1}{2}\|A_{\bar{I}}^\top \bar{\boldsymbol{x}}^{(t)} - \boldsymbol{g}_{\bar{I}}\|^2 + \|\Delta_{\bar{I}}\|_2^2 \\ & && \text{(since } -(a+b)^2 \leq -1/2a^2 + b^2\text{)} \\ \leq & - \frac{k}{\beta}\Delta_d^{(t)} + \|\Delta_{\bar{I}}\|_2^2 && \text{(from (A.5))} \\ \leq & - \frac{k}{\beta}\Delta_d^{(t)} + \frac{k}{n^2}R\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2 \end{aligned}$$

With the relation between $A_{I^{(t)}}^\top \boldsymbol{x}^{(t)} - \boldsymbol{g}_{I^{(t)}}$ and $\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}$ we finish the proof. □

A.1.5 Convergence on Duality Gap

Now we are able to merge the primal/dual progress to get the overall progress on the duality gap.

Proof of Theorem 2.5.1. We simply blend Lemma A.1.2 and Lemma A.1.5 with the primal-dual progress (Lemma A.1.3):

$$
\begin{aligned}
&\Delta_d^{(t)} - \Delta_d^{(t-1)} + \Delta_p^{(t)} - \Delta_p^{(t-1)} \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \frac{1}{2\delta}\|\boldsymbol{y}^{(t)} - \boldsymbol{y}^{(t-1)}\|^2 \\
&+ 2\delta Rk\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2 && \text{(Lemma A.1.3)} \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) + \frac{\delta}{2}(-\frac{k}{\beta}\Delta_d^{(t)} + kR\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2) \\
&+ 2\delta Rk\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2 && \text{(Lemma A.1.5)} \\
= &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \frac{k\delta}{2\beta}\Delta_d^{(t)} + \frac{5R\delta k}{2}\|\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|_2^2 \\
\leq &\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \frac{k\delta}{2\beta}\Delta_d^{(t)} + \frac{5R\delta k}{\mu}(\mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})) \\
= &(1 - \frac{5R\delta k}{\mu})(\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)})) - \frac{k\delta}{2\beta}\Delta_d^{(t)} \\
&+ \frac{5R\delta k}{\mu n^2}(\mathcal{L}(\boldsymbol{x}^{(t+1)}, \boldsymbol{y}^{(t)}) - \mathcal{L}(\bar{\boldsymbol{x}}^{(t)}, \boldsymbol{y}^{(t)})) \\
\leq &-\frac{k\delta}{2\beta}\Delta_d^{(t)} - \left((1 - \frac{5R\delta k}{\mu n^2})\frac{\mu}{4L} - \frac{5R\delta k}{\mu}\right)\Delta_p^{(t)} && \text{(Lemma A.1.2)}
\end{aligned}
$$

When setting $\frac{k\delta}{2\beta} = (1 - \frac{5R\delta k}{\mu n^2})\frac{\mu}{4L} - \frac{5R\delta k}{\mu}$, we get that $\Delta^{(t)} \leq \frac{1}{1+a}\Delta^{(t-1)}$, where $1/a = \mathcal{O}(\frac{L}{\mu}(1 + \frac{R\beta}{\mu}))$. Therefore it takes $\mathcal{O}(\frac{L}{\mu}(1 + \frac{R\beta}{\mu})\log\frac{1}{\epsilon})$ for $\Delta^{(t)}$ to reach ϵ.

When $\beta > 2\alpha$, we could redefine the primal-dual process as $\Delta^{(t)} := (\frac{\beta}{\alpha} - 1)\Delta_d^{(t)} + \Delta_p^{(t)}$ and rewrite some of the key steps, especially for the overall primal-dual progress.

$$
\begin{aligned}
&\Delta^{(t)} - \Delta^{(t-1)} \\
= &(\frac{\beta}{\alpha} - 1)(\Delta_d^{(t)} - \Delta_d^{(t-1)}) + \Delta_p^{(t)} - \Delta_p^{(t-1)}
\end{aligned}
$$

$$
\begin{aligned}
=&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})+\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t-1)})\\
&+\frac{\beta}{\alpha}(\Delta_d^{(t)}-\Delta_d^{(t-1)})\\
\leq&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})+(\frac{1}{\delta}+\frac{1}{2\alpha})\|\boldsymbol{y}_{I^{(t)}}^{(t-1)}-\boldsymbol{y}_{I^{(t)}}^{(t)}\|^2\\
&-\frac{\beta}{\alpha}\langle A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)}-\boldsymbol{x}^{(t)}),\boldsymbol{y}_{I^{(t)}}^{(t)}-\boldsymbol{y}_{I^{(t)}}^{(t-1)}\rangle-\frac{\beta}{\alpha}(\frac{1}{\delta}+\frac{1}{2\beta})\|\boldsymbol{y}^{(t)}-\boldsymbol{y}^{(t-1)}\|^2\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(from Eqn. (A.3) and (A.4))}\\
=&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})-\frac{\beta}{\alpha}\langle A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)}-\boldsymbol{x}^{(t)}),\boldsymbol{y}_{I^{(t)}}^{(t)}-\boldsymbol{y}_{I^{(t)}}^{(t-1)}\rangle\\
&-(\frac{\beta}{\alpha}-1)\frac{1}{\delta}\|\boldsymbol{y}^{(t)}-\boldsymbol{y}^{(t-1)}\|^2\\
\leq&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})+\frac{3\beta}{2\alpha}\delta\|A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)}-\boldsymbol{x}^{(t)})\|^2\\
&-(\frac{3\beta}{4\alpha}-1)\frac{1}{\delta}\|\boldsymbol{y}^{(t)}-\boldsymbol{y}^{(t-1)}\|^2\qquad\qquad\text{(since } ab\leq\delta a^2+1/(4\delta)b^2\text{)}\\
\leq&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})+\frac{\beta}{\alpha}\delta\|A_{I^{(t)},:}(\bar{\boldsymbol{x}}^{(t)}-\boldsymbol{x}^{(t)})\|^2\\
&-\frac{\beta}{4\alpha\delta}\|\boldsymbol{y}^{(t)}-\boldsymbol{y}^{(t-1)}\|^2\qquad\qquad\text{(since } \beta/\alpha\geq 2\text{)}\\
\leq&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})-\frac{\beta}{4\alpha\delta}\|\boldsymbol{y}^{(t)}-\boldsymbol{y}^{(t-1)}\|^2\\
&+\frac{\beta\delta Rk}{\alpha}\|\bar{\boldsymbol{x}}^{(t)}-\boldsymbol{x}^{(t)}\|^2
\end{aligned}
$$

Similarly to the previous setting, we get the whole primal-dual progress is bounded as follows:

$$
\begin{aligned}
&(\frac{\beta}{\alpha}-1)(\Delta_d^{(t)}-\Delta_d^{(t-1)})+\Delta_p^{(t)}-\Delta_p^{(t-1)}\\
\leq&\mathcal{L}(\boldsymbol{x}^{(t+1)},\boldsymbol{y}^{(t)})-\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})-\frac{\beta\delta}{4\alpha}\frac{k}{\beta}\Delta_d^{(t)}\\
&+\frac{5\beta R\delta k}{2\alpha\mu}(\mathcal{L}(\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)})-\mathcal{L}(\bar{\boldsymbol{x}}^{(t)},\boldsymbol{y}^{(t)}))\\
\leq&-\frac{\beta}{4\alpha}\frac{k\delta}{\beta}\Delta_d^{(t)}-\left((1-\frac{5\beta R\delta k}{2\alpha\mu})\frac{\mu}{4L}-\frac{5\beta R\delta k}{2\alpha\mu}\right)\Delta_p^{(t)}
\end{aligned}
$$

Therefore, when we set a proper k and δ such that $\frac{\beta}{4\alpha}\frac{k\delta}{\beta}=(\frac{\beta}{\alpha}-1)\left((1-\frac{5\beta R\delta k}{2\alpha\mu})\frac{\mu}{4L}-\frac{5\beta R\delta k}{2\alpha\mu}\right)$, and

since $\frac{\beta}{\alpha} - 1 \geq \frac{\beta}{2\alpha}$, we get $\delta = \frac{1}{k}(\frac{L}{\mu\beta} + \frac{5\beta R}{2\alpha\mu}(1 + 4\frac{L}{\mu}))^{-1}$. And we have $\Delta^{(t)} - \Delta^{(t-1)} \leq -1/a\Delta^{(t)}$, where $a = \mathcal{O}(\frac{L}{\mu}(1 + \frac{\beta}{\alpha}\frac{R\beta}{\mu}))$. Therefore it takes $t = \mathcal{O}(\frac{L}{\mu}(1 + \frac{\beta}{\alpha}\frac{R\beta}{\mu})\log\frac{1}{\epsilon})$ iterations for the duality gap $\Delta^{(t)}$ to reach ϵ error. □

A.1.6 Smooth Hinge Loss and Relevant Properties

Smooth hinge loss is defined as follows:

$$h(z) = \begin{cases} \frac{1}{2} - z & \text{if } z < 0 \\ \frac{1}{2}(1-z)^2 & \text{if } z \in [0,1] \\ 0 & \text{otherwise.} \end{cases} \tag{A.6}$$

Our loss function over a prediction p associated with a label $\ell_i \in \{\pm 1\}$ will be $f_i(p) = h(p\ell_i)$. The derivative of smooth hinge loss h is:

$$h'(z) = \begin{cases} -1 & \text{if } z < 0 \\ z - 1 & \text{if } z \in [0,1] \\ 0 & \text{otherwise.} \end{cases} \tag{A.7}$$

Its convex conjugate is:

$$h^*(z^*) = \begin{cases} \frac{1}{2}(z^*)^2 + z^* & \text{if } z^* \in [-1,0] \\ \infty & \text{otherwise.} \end{cases} \tag{A.8}$$

Notice since $f_i(p) = h(\ell_i p)$, $f_i^*(p) = h^*(p/\ell_i) = h^*(p\ell_i)$.

Claim A.1.6. *For a convex and β-smooth scalar function f, if it is α strongly convex over some convex set, and linear otherwise, then its conjugate function f^* is $1/\beta$-strongly convex, and it is a $1/\alpha$-smooth function plus an indicator function over some interval $[a, b]$.*

Proof. To begin with, since $f''(x) \leq \beta, \forall x$, meaning f is β-smooth, then with duality we have f^* is $1/\beta$ strongly convex [76]. Secondly, since f is α strongly convex over a convex set, meaning an interval for $\mathbb{R}$, therefore f could only be linear on $(-\infty, a]$ or $[b, \infty)$, and is α-strongly convex

over the set $[a, b]$ (Here for simplicity $a < b$ could be $\pm\infty$). We denote $f'(-\infty) := \lim_{x\to-\infty} f'(x)$ and $f'(-\infty)$ likewise. It's easy to notice that $f'(-\infty) \le f'(a) < f'(b) \le f'(\infty)$ since f is convex overall and strongly convex over $[a, b]$. Therefore $f(y) > f(a) + f'(a)(y - a)$ when $y > a$ and $f(y) = f(a) + f'(a)(y - a)$ when $y \le a$.

Now since $f^*(x^*) \equiv \max_x\{x^*x - f(x)\}$, it's easy to verify that when $x^* < f'(a)$, $x^*x - f(x) = x^*x - f(a) - f'(a)(x - a) = -(f'(a) - x^*)x - f(a) + f'(a)a \to \infty$ when $x \to -\infty$. Similarly, when $x^* > f'(b)$, $f^*(x^*) = \infty$. On the other hand, when $x^* \in [f'(a), f'(b)]$, $f^*(x^*) = \max_x\{x^*x - f(x)\} = \max_{x\in[a,b]}\{x^*x - f(x)\}$. This is because $x^*a - f(a) \ge x^*y - f(y) = x^*y - f(y) - f'(a)(y - a), \forall y \le a$, and similarly $x^*b - f(b) \ge x^*y - f(y)\forall y > b$. Therefore f^* is $1/\alpha$ smooth over the interval $[f'(a), f'(b)]$, where $-\infty \le f'(a) < f'(b) \le \infty$.

□

A.1.7 Convergence of Optimization over Trace Norm Ball

The convergence analysis for trace norm ball is mostly similar to the case of ℓ_1 ball. The most difference lies on the primal part, where our approximated update incur linear progress as well as some error.

Lemma A.1.7 (Primal Progress for Algorithm 2)**.** *Suppose rank $\bar{X}^{(t)} \le s$ and $\epsilon > 0$. If each $\tilde{X}$ computed in our algorithm is a $(\frac{1}{2}, \frac{\epsilon}{8})$-approximate solution to (2.15), then for every t, it satisfies $\mathcal{L}(X^{(t+1)}, Y^{(t)}) - \mathcal{L}(X^{(t)}, Y^{(t)}) \le -\frac{\mu}{8L}\Delta_p^{(t)} + \frac{\epsilon}{16}$.*

Proof. Refer to the proof in [5] we have:

$$\mathcal{L}(X^{(t+1)}, Y^{(t)}) - \mathcal{L}(\bar{X}^{(t)}, Y^{(t)}) \le (1 - \frac{\mu}{8L})\left(\mathcal{L}(X^{(t)}, Y^{(t)})) - \mathcal{L}(\bar{X}^{(t)}, Y^{(t)})\right) + \frac{\epsilon\mu}{16L}$$

Now move the first term on the RHS to the left and rearrange we get:

$$(1-\frac{\mu}{8L})(\mathcal{L}(X^{(t+1)},Y^{(t)})-\mathcal{L}(X^{(t)},Y^{(t)}))+\frac{\mu}{8L}\left(\mathcal{L}(X^{(t+1)},Y^{(t)}))-\mathcal{L}(\bar{X}^{(t)},Y^{(t)})\right)\leq\frac{\epsilon\mu}{16L}$$

Therefore we get:

$$\mathcal{L}(X^{(t+1)},Y^{(t)})-\mathcal{L}(X^{(t)},Y^{(t)}))\leq-\frac{\mu}{8L}\Delta_p^{(t)}+\frac{\epsilon}{16}.$$

□

Now back to the convergence guarantees on the trace norm ball.

Proof of Theorem 2.5.4. We again define $\Delta=A(\bar{X}^{(t)}-X^{(t)})$. $G=\nabla_Y\mathcal{L}(X^{(t)},Y^{(t)})$ such that $Y^{(t)}_{I^{(t)},:}-Y^{(t-1)}_{I^{(t)},:}=\delta(\langle A_{I^{(t)},:}X^{(t)}\rangle-G_{I^{(t)},:})$. Again we get $\|\Delta\|_F^2\leq R\|\bar{X}^{(t)}-X^{(t)}\|_F^2$.

$$\Delta_d^{(t)}\leq\frac{\beta}{2}\|A\bar{X}^{(t)}-G\|_F^2\leq\frac{n\beta}{2k}\|A_{I^{(t)},:}\bar{X}^{(t)}-G_{I^{(t)},:}\|_F^2$$

Other parts are exactly the same and we get:

$$\begin{aligned}
&(\frac{\beta}{\alpha}-1)(\Delta_d^{(t)}-\Delta_d^{(t-1)})+\Delta_p^{(t)}-\Delta_p^{(t-1)}\\
\leq&\mathcal{L}(X^{(t+1)},Y^{(t)})-\mathcal{L}(X^{(t)},Y^{(t)})-\frac{\beta\delta}{4\alpha}\frac{k}{\beta}\Delta_d^{(t)}\\
&+\frac{5\beta R\delta k}{2\alpha\mu}(\mathcal{L}(X^{(t)},Y^{(t)})-\mathcal{L}(\bar{X}^{(t)},Y^{(t)}))\\
\leq&-\frac{\beta}{4\alpha}\frac{k\delta}{\beta}\Delta_d^{(t)}-\left((1-\frac{5\beta R\delta k}{2\alpha\mu})\frac{\mu}{8L}-\frac{5\beta R\delta k}{2\alpha\mu}\right)\Delta_p^{(t)}+(1-\frac{5\beta R\delta k}{2\alpha\mu})\frac{\epsilon}{16}
\end{aligned}$$

(Lemma A.1.7)

Therefore when $\delta\leq\frac{1}{k}(\frac{L}{\mu\beta}+\frac{5\beta R}{2\alpha\mu}(1+8\frac{L}{\mu}))^{-1}$, it satisfies $\Delta^{(t)}-\Delta^{(t-1)}\leq-\frac{k\delta}{2\beta}\Delta^{(t)}+\frac{\epsilon}{16}$. Therefore denote $a=\frac{2\beta}{k\delta}$, we get $\Delta^{(t)}\leq\frac{a}{a+1}(\Delta^{(t-1)}+\frac{\epsilon}{16})$. Therefore we get $\Delta^{(t)}\leq(\frac{a}{a+1})^t\Delta^{(0)}+$

$\frac{\epsilon}{16}\sum_{i=1}^{t}(\frac{a}{a+1})^i \leq (\frac{c}{c+1})^t\Delta^{(0)}+\epsilon/16$. Since $(\frac{a}{a+1})^t \leq e^{-t/a}$, it takes around $a = \mathcal{O}(\frac{L}{\mu}(1+\frac{\beta}{\alpha}\frac{R\beta}{\mu})\log\frac{1}{\epsilon})$ iterations for the duality gap to get ϵ-error. □

A.1.8 Difficulty on Extension to Polytope Constraints

Another important type of constraint we have not explored in this paper is the polytope constraint. Specifically,

$$\min_{\boldsymbol{x}\in M\subset\mathbb{R}^d} f(A\boldsymbol{x})+g(\boldsymbol{x}), M = conv(\mathcal{A}), \text{with only access to: LMO}_{\mathcal{A}(\boldsymbol{r})} \in \arg\min_{\boldsymbol{x}\in\mathcal{A}}\langle \boldsymbol{r},\boldsymbol{x}\rangle,$$

where $\mathcal{A} \subset \mathbb{R}^d, |\mathcal{A}| = m$ is a finite set of vectors that is usually referred as atoms. It is worth noticing that this linear minimization oracle (LMO) for FW step naturally chooses a single vector in $\mathcal{A}$ that minimizes the inner product with $\boldsymbol{x}$. Again, this FW step creates some "partial update" that could be appreciated in many machine learning applications. Specifically, if our computation of gradient is again dominated by a matrix-vector (data matrix versus variable $\boldsymbol{x}$) inner product, we could possibly pre-compute each value of $\boldsymbol{v}_i := A\boldsymbol{x}_i, \boldsymbol{x}_i \in \mathcal{A}$, and simply use $\boldsymbol{v}_i$ to update the gradient information when $\boldsymbol{x}_i$ is the greedy direction provided by LMO.

When connecting to our sparse update case, we are now looking for a k-sparse update, $k \ll m = |\mathcal{A}|$, with the basis of $\mathcal{A}$, i.e., $\tilde{\boldsymbol{x}} = \sum_{i=1}^{k}\lambda_i\boldsymbol{x}_{n_i}, \boldsymbol{x}_{n_i} \in \mathcal{A}$. In this way, when we update $\boldsymbol{x}^+ \leftarrow (1-\eta)\boldsymbol{x}+\eta\tilde{\boldsymbol{x}}$, we will only need to compute $\sum_{i=1}^{k}\boldsymbol{v}_{n_i}$ which is $\mathcal{O}(kd)$ time complexity.

However, to enforce such update that is "sparse" on $\mathcal{A}$ is much harder. To migrate our algorithms with ℓ_1 ball or trace norm ball, we will essentially be solving the following problem:

$$\tilde{\boldsymbol{x}} \leftarrow \arg\min_{\Lambda\in\Delta^m, \|\Lambda\|_0\leq k, \boldsymbol{x}=\sum_{i=1}^{m}\lambda_i\boldsymbol{x}_i, \boldsymbol{x}_i\in\mathcal{A}} \langle \boldsymbol{g},\boldsymbol{y}\rangle + \frac{1}{2\eta}\|\boldsymbol{y}-\boldsymbol{x}\|_2^2,$$

where Δ^m is the m dimensional simplex, and $\boldsymbol{g}$ is the current gradient vector.

Unlike the original sparse recovery problem that could be relaxed with an ℓ_1 constraint to softly encourage sparsity, it's generally much harder to find the k sparse Λ in this case. Actually, it is as hard as the lattice problem [82] and is NP hard in general.

Therefore we are not able to achieve linear convergence with cheap update with polytope-type constraints. Nonetheless, the naive FW with primal dual formulation should still be computational efficient in terms of per iteration cost, where a concentration on SVM on its dual form has been explored by [86].

A.2 Discussions on Efficient Coordinate Selections

The modified Block Frank-Wolfe step in Eqn. (2.6) achieves an s-sparse update of the iterates and could be computed efficiently when one knows which s coordinates to update. However, in order to find the s coordinates, one needs to compute the full gradient $\nabla f(\boldsymbol{x})$ with naive implementation. This phenomenon reminds us of greedy coordinate descent.

Even with the known fact that coordinate descent converges faster with greedy selection than with random order[129], there have been hardness to propogate this idea because of expensive greedy selections since the arguments that GCD converges similarly with RCD in [124], except for special cases [99, 97, 44, 77]. This is also probability why the partial updates nature of FW steps is less exploited before.

We investigate some possible tricks to boost GCD method that could be possibly applied to FW methods. A recent paper [77], Karimireddy et al. make connections between the efficient choice of the greedy coordinates with the problem of Maximum Inner Product Search (MIPS) for a composite function $P(\boldsymbol{x}) = f(A\boldsymbol{x}) + g(\boldsymbol{x})$, where $A \in \mathbb{R}^{n\times d}$. We rephrase the connection for the Frank-Wolfe algorithm. Since the computation of gradient is essentially $A^\top \nabla f_{|A\boldsymbol{x}} + \nabla g(\boldsymbol{x})$, to find its largest magnitude is to search maximum inner products among:

$$\pm\langle[\tilde{\boldsymbol{a}}_i^\top|1], [\nabla f_{|A\boldsymbol{x}}^\top|\nabla_i g(\boldsymbol{x})]\rangle, \text{i.e.} \pm \left(\tilde{\boldsymbol{a}}_i^\top \nabla f_{|A\boldsymbol{x}} + \nabla_i g(\boldsymbol{x})\right),$$

where $\tilde{\boldsymbol{a}}_i \in \mathbb{R}^n$ is the i-th column of data matrix A, and $\nabla f_{|A\boldsymbol{x}}$ is the gradient of f at $A\boldsymbol{x}$. In this way, we are able to select the greedy coordinates by conducting MIPS for a fixed $\mathbb{R}^{2d\times(n+1)}$ matrix $[A^\top|I| - A^\top| - I]^\top$ and each newly generated vector $[\nabla f_{|A\boldsymbol{x}}^\top|\nabla g_i(\boldsymbol{x})]$. Therefore when ∇g_i is constant for linear function or $\pm\lambda$ for $g(\boldsymbol{x}) = \lambda\|\boldsymbol{x}\|_1$, we could find the largest magnitude of the gradient in sublinear time. Still, the problems it could conquer is very limited. It doesn't even work

for ℓ_2 regularizer since the different coordinates in $\nabla_i g(\boldsymbol{x})$ creates d new vectors in each iteration and traditional MIPS could resolve it in time sublinear to d. Meanwhile, even with constant $\nabla_i g(\boldsymbol{x})$, it still requires at least $\mathcal{O}((2d)^c \log(d))$ times of inner products of dimension $n+1$ for some constant c [147].

However, we have shown that for general composite form $f(A\boldsymbol{x}) + g(\boldsymbol{x})$ with much more relaxed requirements on the regularizer g, we are able to select and update each coordinate with *constant* times of inner products on average while achieving linear convergence. Therefore the usage of these tricks applied on FW method (MIPS as well as the nearest neighbor search [44]) is completely dominated by our contribution and we omit them in the main text of this paper.

A.3 More Results on Empirical Studies

A.3.1 More experiments with ℓ_1 norm

To investigate more on how our algorithms perform with different choices of parameters, we conducted more empirical studies with different settings of condition numbers. Specifically, we vary the parameter μ that controls the strong convexity of the primal function. Experiments are shown in Figure A.1.

A.3.2 Experiments with trace norm ball on synthetic data

For trace norm constraints, we also implemented our proposal Primal Dual Block Frank Wolfe to compare with some prior work, especially Block FW [5]. Since prior work were mostly implemented in Matlab to tackle trace norm projections, we therefore also use Matlab to show fair comparisons. We choose quadratic loss $f(AX) = \|AX - B\|_F^2$ and g to be ℓ_2 regularizer with $\mu = 10/n$. The synthetic sensing matrix $A \in \mathbb{R}^{n \times d}$ is dense with $n = 1000$ and $d = 800$. Our

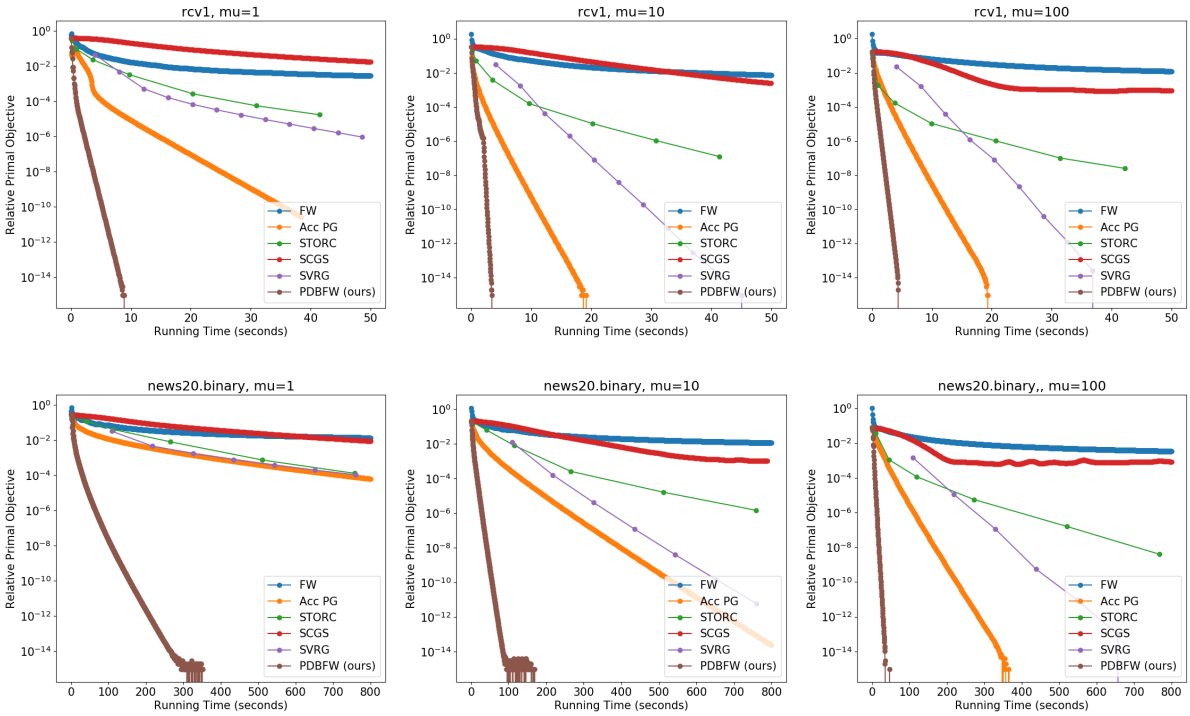


Figure A.1: Convergence result comparison of different algorithms on smoothed hinge loss by varying the coefficient of the regularizer. The first row is the results ran on the rcv1.binary dataset, while the second row is the results ran on the news20.binary dataset. The first column is the result when the regularizer coeffcient μ is set to $1/n$. The middle column is when $\mu = 10/n$, and the right column is when $\mu = 100/n$.

observation B is of dimension 1000×600 and is generated by a ground truth matrix X_0 such that $B = AX_0$. Here $X_0 \in \mathbb{R}^{800\times 600}$ is constructed with low rank structure. We vary its rank s to be $10, 20,$ and 100. The comparisons with stochastic FW, blockFW [5], STORC [66], SCGS [87], and projected SVRG [74] are presented in Figure A.2, which verifies that our proposal PDBFW consistently outperforms the baseline algorithms.

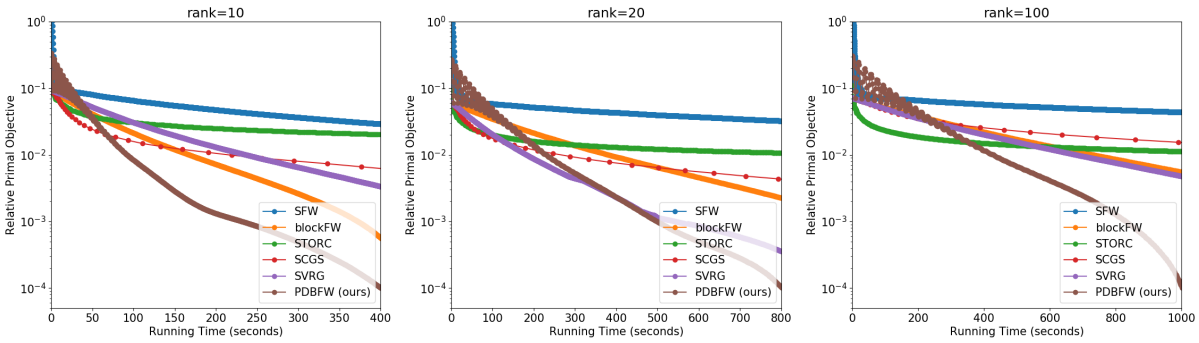


Figure A.2: Convergence comparison of our Primal Dual Block Frank Wolfe and other baselines. Figures show the relative primal objective value decreases with the wall time.

Appendix B

Appendix for Optimistic Multiplicative Weight Update

B.1 Equations of the Jacobian of OMWU

$$\frac{\partial g_{1,i}}{\partial x_i} = \frac{e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}}{S_x} + x_i\frac{1}{S_x^2}\left(e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}(-2\eta\frac{\partial^2 f}{\partial x_i^2})S_x - e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\frac{\partial S_x}{\partial x_i}\right) \tag{B.1}$$

$$\text{where } \frac{\partial S_x}{\partial x_i} = e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}} - 2\eta\sum_k x_k e^{-2\eta\frac{\partial f}{\partial x_k}+\eta\frac{\partial f}{\partial z_k}}\frac{\partial^2 f}{\partial x_i^2} \tag{B.2}$$

$$\frac{\partial g_{1,i}}{\partial x_j} = x_i\frac{1}{S_x^2}\left(e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}(-2\eta\frac{\partial^2 f}{\partial x_i\partial x_j})S_x - e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\frac{\partial S_x}{\partial x_j}\right) \tag{B.3}$$

$$\text{where } \frac{\partial S_x}{\partial x_j} = e^{-2\eta\frac{\partial f}{\partial x_j}+\eta\frac{\partial f}{\partial z_j}} - 2\eta\sum_k x_k e^{-2\eta\frac{\partial f}{\partial x_k}+\eta\frac{\partial f}{\partial z_k}}\frac{\partial^2 f}{\partial x_j\partial x_k} \tag{B.4}$$

$$\frac{\partial g_{1,i}}{\partial y_j} = x_i\frac{1}{S_x^2}\left(e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}(-2\eta\frac{\partial^2 f}{\partial x_i\partial y_j})S_x - e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\frac{\partial S_x}{\partial y_j}\right) \tag{B.5}$$

$$\text{where } \frac{\partial S_x}{\partial y_j} = \sum_k x_k e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}(-2\eta\frac{\partial^2 f}{\partial x_k\partial y_j}) \tag{B.6}$$

$$\frac{\partial g_{1,i}}{\partial z_j} = x_i\frac{1}{S_x^2}\left(e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}(\eta\frac{\partial^2 f}{\partial z_j\partial x_i})S_x - e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\frac{\partial S_x}{\partial z_j}\right) \tag{B.7}$$

$$\text{where } \frac{\partial S_x}{\partial z_j} = \eta\sum_k x_k e^{-2\eta\frac{\partial f}{\partial x_k}+\eta\frac{\partial f}{\partial z_k}}\frac{\partial^2 f}{\partial z_k\partial z_j} \tag{B.8}$$

$$\frac{\partial g_{1,i}}{\partial w_j} = x_i\frac{1}{S_x^2}\left(e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\eta\frac{\partial^2 f}{\partial z_i\partial w_j}S_x - e^{-2\eta\frac{\partial f}{\partial x_i}+\eta\frac{\partial f}{\partial z_i}}\frac{\partial S_x}{\partial w_j}\right) \tag{B.9}$$

$$\text{where } \frac{\partial S_x}{\partial w_j} = \sum_k x_k e^{-2\eta\frac{\partial f}{\partial x_k}+\eta\frac{\partial f}{\partial z_k}}\eta\frac{\partial f}{\partial z_k\partial w_j} \tag{B.10}$$

$$\frac{\partial g_{2,i}}{\partial x_j} = y_i \frac{1}{S_y^2} \left(e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} (2\eta \frac{\partial^2 f}{\partial x_j \partial y_i}) S_y - e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} \frac{\partial S_y}{\partial x_j} \right) \tag{B.11}$$

$$\text{where } \frac{\partial S_y}{\partial x_j} = \sum_k y_k e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} 2\eta \frac{\partial^2 f}{\partial x_j \partial y_k} \tag{B.12}$$

$$\frac{\partial g_{2,i}}{\partial y_i} = \frac{e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}}}{S_y} + y_i \frac{1}{S_y^2} \left(e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} 2\eta \frac{\partial^2 f}{\partial y_i^2} S_y - e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} \frac{\partial S_y}{\partial y_i} \right) \tag{B.13}$$

$$\text{where } \frac{\partial S_y}{\partial y_i} = e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} + 2\eta \sum_k y_k e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} \frac{\partial^2 f}{\partial y_i \partial y_k} \tag{B.14}$$

$$\frac{\partial g_{2,i}}{\partial z_j} = y_i \frac{1}{S_y^2} \left(e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} (-\eta \frac{\partial^2 f}{\partial w_i \partial z_j}) S_y - e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} \frac{\partial S_y}{\partial z_j} \right) \tag{B.15}$$

$$\text{where } \frac{\partial S_y}{\partial z_j} = \sum_k y_k e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} (-\eta \frac{\partial^2 f}{\partial w_k \partial z_j}) \tag{B.16}$$

$$\frac{\partial g_{2,i}}{\partial w_j} = y_i \frac{1}{S_y^2} \left(e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} (-\eta \frac{\partial^2 f}{\partial w_i \partial w_j}) - e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} \frac{\partial S_y}{\partial w_j} \right) \tag{B.17}$$

$$\text{where } \frac{\partial S_y}{\partial w_j} = \sum_k y_k e^{2\eta \frac{\partial f}{\partial y_i} - \eta \frac{\partial f}{\partial w_i}} (-\eta \frac{\partial^2 f}{\partial w_k \partial w_j}) \tag{B.18}$$

B.2 Equations of the Jacobian of OMWU at the fixed point $(\vec{x}^*, \vec{y}^*, \vec{z}^*, \vec{w}^*)$

In this section, we compute the equations of the Jacobian at the fixed point $(\vec{x}^*, \vec{y}^*, \vec{z}^*, \vec{w}^*)$. The fact that $(\vec{x}^*, \vec{y}^*) = (\vec{z}^*, \vec{w}^*)$ and $(\vec{z}, \vec{w})$ takes the position of $(\vec{x}, \vec{y})$ in computing partial derivatives gives the following equations.

$$\frac{\partial g_{1,i}}{\partial x_i} = 1 - x_i^* - 2\eta x_i^*(\frac{\partial^2 f}{\partial x_i^*} - \sum_k x_k^* \frac{\partial^2 f}{\partial x_i \partial x_k}), i \in [n], \tag{B.19}$$

$$\frac{\partial g_{1,i}}{\partial x_j} = -x_i^* - 2\eta x_i^*(\frac{\partial^2 f}{\partial x_i \partial x_j} - \sum_k x_k^* \frac{\partial^2 f}{\partial x_j \partial x_k}), j \in [n], j \neq i \tag{B.20}$$

$$\frac{\partial g_{1,i}}{\partial y_j} = -2\eta x_i^*(\frac{\partial^2 f}{\partial x_i \partial y_j} - \sum_k x_k^* \frac{\partial^2 f}{\partial x_k \partial y_j}), j \in [m] \tag{B.21}$$

$$\frac{\partial g_1, i}{\partial z_j} = \eta x_i^*(\frac{\partial^2 f}{\partial x_i \partial x_j} - \sum_k x_k^* \frac{\partial^2 f}{\partial x_k \partial x_j}), j \in [n] \tag{B.22}$$

$$\frac{\partial g_{1,i}}{\partial w_j} = \eta x_i^*(\frac{\partial^2 f}{\partial x_i \partial y_j} - \sum_k x_k^* \frac{\partial^2 f}{\partial x_k \partial y_j}), j \in [m] \tag{B.23}$$

$$\frac{\partial g_{2,i}}{\partial x_j} = 2\eta y_i^*(\frac{\partial^2 f}{\partial x_j \partial y_i} - \sum_k y_k^* \frac{\partial^2 f}{\partial x_j \partial y_k}), j \in [n] \tag{B.24}$$

$$\frac{\partial g_{2,i}}{\partial y_i} = 1 - y_i^* + 2\eta(\frac{\partial^2 f}{\partial y_i^2} - \sum_k y_k^* \frac{\partial^2 f}{\partial y_i \partial y_k}), i \in [m] \tag{B.25}$$

$$\frac{\partial g_{2,i}}{\partial y_j} = -y_i^* + 2\eta(\frac{\partial^2 f}{\partial y_i \partial y_j} - \sum_k y_k^* \frac{\partial^2 f}{\partial y_j \partial y_k}), j \in [m] \tag{B.26}$$

$$\frac{\partial g_{2,i}}{\partial z_j} = \eta y_i^*(-\frac{\partial^2 f}{\partial x_j \partial y_i} + \sum_k y_k^* \frac{\partial^2 f}{\partial x_j \partial y_k}), j \in [n] \tag{B.27}$$

$$\frac{\partial g_{2,i}}{\partial w_j} = \eta y_i^*(-\frac{\partial^2 f}{\partial y_i \partial y_j} + \sum_k y_k^* \frac{\partial^2 f}{\partial y_k \partial y_j}), j \in [m] \tag{B.28}$$

$$\frac{\partial g_{3,i}}{\partial x_i} = 1 \text{ for all } i \in [n] \text{ and zerofor all the other partial derivatives of } g_{3,i} \tag{B.29}$$

$$\frac{\partial g_{4,i}}{\partial y_i} = 1 \text{ for all } i \in [m] \text{ and zero for all the other partial derivatives of } g_{4,i}. \tag{B.30}$$

B.3 Jacobian matrix at $(\vec{x}^*, \vec{y}^*, \vec{z}^*, \vec{w}^*)$

This section serves for the "Spectral Analysis" of Section 3. The Jacobian matrix of g at the fixed point is obtained based on the calculations above. We refer the main article for the subscript indicating the size of each block matrix.

$$J = \begin{bmatrix} \vec{I} - D_{\vec{x}^*}\vec{1}\vec{1}^\top - 2\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}}f & -2\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{y}}f & \eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}}f & \eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{y}}f \\ 2\eta D_{\vec{y}^{*\top}}(\vec{I} - \vec{1}\vec{y}^*)\nabla^2_{\vec{y}\vec{x}}f & \vec{I} - D_{\vec{y}^*}\vec{1}\vec{1}^\top + 2\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}}f & -\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{x}}f & -\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}}f \\ \vec{I} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{I} & \vec{0} & \vec{0} \end{bmatrix}$$

By acting on the tangent space of each simplex, we observe that $D_{\vec{x}^*}\vec{1}\vec{1}^\top\vec{v} = 0$ for $\sum_k v_k = 0$, so each eigenvalue of matrix J is an eigenvalue of the following matrix

$$J_{\text{new}} = \begin{bmatrix} \vec{I} - 2\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}}f & -2\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{y}}f & \eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}}f & \eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{y}}f \\ 2\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{x}}f & \vec{I} + 2\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}}f & -\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{x}}f & -\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}}f \\ \vec{I} & \vec{0} & \vec{0} & \vec{0} \\ \vec{0} & \vec{I} & \vec{0} & \vec{0} \end{bmatrix}$$

The characteristic polynomial of J_{new} is $\det(J_{new} - \lambda I)$ that can be computed as the determinant of the following matrix:

$$\begin{bmatrix} (1-\lambda)\vec{I} + (\frac{1}{\lambda} - 2)\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{x}}f & (\frac{1}{\lambda} - 2)\eta D_{\vec{x}^*}(\vec{I} - \vec{1}\vec{x}^{*\top})\nabla^2_{\vec{x}\vec{y}}f \\ (2 - \frac{1}{\lambda})\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{x}}f & (1-\lambda)\vec{I} + (2 - \frac{1}{\lambda})\eta D_{\vec{y}^*}(\vec{I} - \vec{1}\vec{y}^{*\top})\nabla^2_{\vec{y}\vec{y}}f \end{bmatrix} \tag{B.31}$$

Appendix C

Appendix for Learning One-layer Generative Model

C.1 Omitted Proof for Hardness

Proof of Theorem 4.3.1. We consider the problem:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \phi(-A\boldsymbol{x} + 2\mathbb{1})^\top \boldsymbol{y}_1 + (\phi(\mathbb{1}^\top \boldsymbol{x}) + \phi(-\mathbb{1}^\top \boldsymbol{x}) - n)y_2 + \phi(\boldsymbol{x} - 1)^\top \boldsymbol{y}_3 + \phi(-\boldsymbol{x} - 1)^\top \boldsymbol{y}_4.$$

It could be easily verified that f falls into the problem set we consider with proper stacking of $\boldsymbol{y}_1, \boldsymbol{y}_3, \boldsymbol{y}_4$ and scalar y_2. We write it in this form for the ease for interpretation and reduction proof. First, notice if there exists a stationary point $\boldsymbol{x}^*, \boldsymbol{y}^*$, $\nabla_{\boldsymbol{y}} f(\boldsymbol{x}^*, \boldsymbol{y}^*) = 0$. Therefore each term on $\boldsymbol{x}$ should be 0. One on hand, the last two terms $\phi(-\boldsymbol{x}^* - 1) = 0$ and $\phi(\boldsymbol{x}^* - 1) = 0$ makes sure that $x_i^* \in [-1, 1]$. Then the second term that guarantees $\sum_i |x_i^*| = n$ means x_i^* could only take binary values. Finally notice any 3SAT problem could be written as a matrix $A \in \mathbb{R}^{m \times d}$ where each row is 3-sparse and binary, and $\boldsymbol{a}_i$ dot product with a binary vector could only take the value of $-3, -1, 1, 3$. And if the value is greater or equal to -2, it means the corresponding clause is satisfied. In fact, we note that $\phi(-A\boldsymbol{x}^* + 2\mathbb{1}) = 0$ means that $A\boldsymbol{x}^* \geq -2$ meaning each conjunction is satisfied. Therefore checking if there exists a stationary point is equivalent to answer the question whether 3SAT is satisfiable.

□

C.2 Omitted Proof for Learning the Distribution

C.2.1 Stationary Point for Matching First Moment

Proof of Lemma 4.4.1. To start with, we consider odd-plus-constant monotone increasing activations. Notice that by proposing a rectified linear discriminator, we have essentially modified the activation function as $\tilde{\phi} := R(\phi - C)$, where $C = \frac{1}{2}(\phi(x) + \phi(-x))$ is the constant bias term of ϕ.

Observe that we can rewrite the objective $\bar{f}_1$ for this case as follows:

$$f_1(A, \boldsymbol{v}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k_0\times k_0})}\, \boldsymbol{v}^\top \tilde{\phi}(A^*\boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, \boldsymbol{v}^\top \tilde{\phi}(A\boldsymbol{z}).$$

Moreover, notice that $\tilde{\phi}$ is positive and increasing on its support which is $[0, +\infty)$.

Now let us consider the other case in our statement where ϕ has a positive and monotone increasing even component in $[0, +\infty)$. In this case, let us take:

$$\tilde{\phi}(x) = \begin{cases} \phi(x) + \phi(-x), & x \geq 0 \\ 0, & \text{o.w.} \end{cases}$$

Because of the symmetry of the Gaussian distribution, we can rewrite the objective function for this case as follows:

$$f_1(A, \boldsymbol{v}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k_0\times k_0})}\, \boldsymbol{v}^\top \tilde{\phi}(A^*\boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, \boldsymbol{v}^\top \tilde{\phi}(A\boldsymbol{z}).$$

Moreover, notice that $\tilde{\phi}$ is positive and increasing on its support which is $[0, +\infty)$.

To conclude, in both cases, the optimization objective can be written as follows, where $\tilde{\phi}$ satisfies Assumption 4.4.1.2 and is only non-zero on $[0, +\infty)$.

$$f_1(A, \boldsymbol{v}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k_0\times k_0})}\, \boldsymbol{v}^\top \tilde{\phi}(A^*\boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, \boldsymbol{v}^\top \tilde{\phi}(A\boldsymbol{z}).$$

The stationary points of the above objective satisfy:

$$\begin{cases} \nabla_{\boldsymbol{v}} f_1(A, \boldsymbol{v}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k_0\times k_0})}\, \tilde{\phi}(A^*\boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, \tilde{\phi}(A\boldsymbol{z}) = 0, \\ \nabla_{\boldsymbol{a}_j} f_1(A, \boldsymbol{v}) = -\,\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, v_j \tilde{\phi}'(\boldsymbol{a}_j^\top \boldsymbol{z})\boldsymbol{z} = 0. \end{cases}$$

We focus on the gradient over $\boldsymbol{v}$. To achieve $\nabla_{\boldsymbol{v}} f_1(A, \boldsymbol{v}) = 0$, the stationary point satisfies:

$$\forall j, \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k_0\times k_0})}\, \tilde{\phi}((\boldsymbol{a}_j^*)^\top \boldsymbol{z}) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\, \tilde{\phi}(\boldsymbol{a}_j^\top \boldsymbol{z}), \text{ i.e.}$$

$$\forall j, \mathbb{E}_{x\sim\mathcal{N}(0,\|\boldsymbol{a}_j^*\|^2)}\,\tilde{\phi}(x) = \mathbb{E}_{x'\sim\mathcal{N}(0,\|\boldsymbol{a}_j\|^2)}\,\tilde{\phi}(x'). \tag{C.1}$$

To recap, for activations ϕ that follow Assumption 4.4.1, in both cases we have written the necessary condition on stationary point to be Eqn. (C.1), where $\tilde{\phi}$ is defined differently for odd or non-odd activations, but in both cases it is positive and monotone increasing on its support $[0, \infty)$. We then argue the only solution for Eqn. (C.1) satisfies $\|\boldsymbol{a}_j\| = \|\boldsymbol{a}_j^*\|, \forall j$. This follows directly from the following claim:

Claim C.2.1. *The function $h(\alpha) := \mathbb{E}_{x\sim\mathcal{N}(0,\alpha^2)}\, f(x), \alpha > 0$ is a monotone increasing function if f is positive and monotone increasing on its support $[0, \infty)$.*

We could see from Claim C.2.1 that the LHS and RHS of Eqn. (C.1) is simply $h(\|\boldsymbol{a}_j\|)$ and $h(\|\boldsymbol{a}_j^*\|)$ for each j. Now that h is an monotone increasing function, the unique solution for $h(\|\boldsymbol{a}_j\|) = h(\|\boldsymbol{a}_j^*\|)$ is to match the norm: $\|\boldsymbol{a}_j\| = \|\boldsymbol{a}_j^*\|, \forall j$.

Proof of Claim C.2.1.

$$\begin{aligned} h(\alpha) &= \mathbb{E}_{x\sim\mathcal{N}(0,\alpha^2)}\, f(x) \\ &= \int_0^\infty f(x) e^{-\frac{x^2}{2\alpha^2}} dx \\ &\overset{y:=x/\alpha}{=} \int_0^\infty \alpha f(\alpha y) e^{-\frac{y^2}{2}} dy \\ &= \mathbb{E}_{y\sim\mathcal{N}(0,1)}\, \alpha f(\alpha y). \end{aligned}$$

Notice $h'(\alpha) = \mathbb{E}_{x\sim\mathcal{N}(0,1)}[\alpha x f'(\alpha x) + f(\alpha x)]$. Since f, f', and $\alpha > 0$, and we only care about the support of f where x is also positive, therefore h' is always positive and h is monotone increasing. □

To sum up, at stationary point where $\nabla f_1(A, \boldsymbol{v}) = 0$, we have

$$\forall i, \|\boldsymbol{a}_i^*\| = \|\boldsymbol{a}_i\|.$$

□

C.2.2 Proof of Theorem 4.4.2

Proof of Theorem 4.4.2. We will take optimal gradient ascent steps with learning rate 1 on the discriminator side $\boldsymbol{v}$, hence the function we will actually be optimizing over becomes (using the notation for $\tilde{\phi}$ from section C.2.1):

$$h(A) = \max_{\boldsymbol{v}} f_1(A, \boldsymbol{v}) = \frac{1}{2} \left\| \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \tilde{\phi}(A^* \boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \tilde{\phi}(A\boldsymbol{z}) \right\|^2.$$

We just want to verify that there's no spurious local minimum for $h(A)$. Notice there's no interaction between each row vector of A. Therefore we instead look at each

$$h_i := \frac{1}{2} \left(\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \tilde{\phi}((\boldsymbol{a}_i^*)^\top \boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \tilde{\phi}(\boldsymbol{a}_i^\top \boldsymbol{z}) \right)^2$$

for individual i. Now $\nabla h_i(\boldsymbol{a}_i) = -\left(\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \tilde{\phi}((\boldsymbol{a}_i^*)^\top \boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \tilde{\phi}(\boldsymbol{a}_i^\top \boldsymbol{z}) \right) \left(\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \boldsymbol{z} \tilde{\phi}'(\boldsymbol{a}_i^\top \boldsymbol{z}) \right)$. Due to the symmetry of the Gaussian, we take $\boldsymbol{a}_i = a\boldsymbol{e}_1$, where $a = \|\boldsymbol{a}_i\|$. It is easy to see that checking whether $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \boldsymbol{z} \tilde{\phi}'(\boldsymbol{a}_i^\top \boldsymbol{z}) = 0$ is equivalent to checking whether $\mathbb{E}_{z_1 \sim \mathcal{N}(0,1)} z_1 \tilde{\phi}'(a z_1) = 0$.

Recall that $\tilde{\phi}$ is supported on $[0, +\infty)$ and it is monotonically increasing on its support. Hence, $\mathbb{E}_{z_1 \sim \mathcal{N}(0,1)} z_1 \tilde{\phi}'(a z_1) \neq 0$ unless $a = 0$. Hence, suppose $\|\boldsymbol{a}_i\| \neq 0, \forall i$. Then $\nabla_A h(A) = 0$ iff $h(A) = 0$, i.e. $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \tilde{\phi}(A^* \boldsymbol{z}) = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \tilde{\phi}(A\boldsymbol{z})$.

Therefore all stationary points of $h(A)$ are global minima where $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \tilde{\phi}(A^* \boldsymbol{z}) = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \tilde{\phi}(A\boldsymbol{z})$ and according to Lemma 4.4.1, this only happens when $\|\boldsymbol{a}_i\| = \|\boldsymbol{a}_i^*\|, \forall i \in [d]$. □

C.2.3 Stationary Points for WGAN with Quadratic Discriminator

Proof of Lemma 4.5.2. To study the stationary point for $\tilde{g}(Z) = \sum_{jk} \tilde{g}_{jk}(z_{jk})$, we look at each individual $\tilde{g}_{jk}(z) \equiv \frac{1}{2}(\sum_{i=0}^{\infty} \sigma_i^2((z_{jk}^*)^i - z^i))^2$.

Notice for odd-plus-constant activations, σ_i is zero for even $i > 0$. Recall our assumption in Lemma 4.5.2 also requires that $\sigma_1 \neq 0$. Since the analysis is invariant to the which entry of matrix Z we are studying, we simplify the notation here and study the stationary points of $f(a) = \frac{1}{2}(\sum_{i \text{ odd}} \sigma_i^2(a^i - b^i))^2$ for some constants b and σ_i, where $\sigma_1 \neq 0$.[1]

$$\begin{aligned} f'(a) &= \left(\sum_{i \text{ odd}} \sigma_i^2(a^i - b^i)\right)\left(\sum_{i \text{ odd}} i\sigma_i^2 a^{i-1}\right) \\ &= (a-b)\left(\sigma_1^2 + \sum_{i \geq 3 \text{ odd}} \sigma_i^2 \frac{a^i - b^i}{a-b}\right)\left(\sigma_1^2 + \sum_{i \geq 3 \text{ odd}} i\sigma_i^2 a^{i-1}\right) \\ &= (a-b)(\text{I})(\text{II}). \end{aligned}$$

Notice now $f'(a) = 0 \Leftrightarrow a = b$. This is because the polynomial $f'(a)$ is factorized to $a - b$ and two factors I and II that are always positive. Notice here we use $\frac{a^i - b^i}{a-b}$ to denote $\sum_{j=0}^{i} a^j b^{i-j}$, which is always nonnegative. This is simply because $a^i - b^i$ always shares the same sign as $a - b$ when i is odd. Therefore I=$\sigma_1^2 + \sum_{i \geq 3 \text{ odd}} \sigma_i^2 \frac{a^i - b^i}{a-b} > 0, \forall a$.

Meanwhile, since a^{i-1} is always nonnegative for each odd i, we have II= $\sigma_1^2 + \sum_{i \geq 3 \text{ odd}} i\sigma_i^2 a^{i-1}$ is also always positive for any a.

Next, for activation like ReLU, loss $\tilde{g}_{jk}(z) = \frac{1}{2}(h(z) - h(z_{jk}^*))^2$, where $h(x) = \frac{1}{\pi}(\sqrt{1-x^2} + (\pi - cos^{-1}(x))x)$ [35]. Therefore $h'(-1) = 0$ for any z_{jk}^*. This fact prevents us from getting the same conclusion for ReLU.

[1] The zero component has been cancelled out.

However, for leaky ReLU with coefficient of leakage $\alpha \in (0,1)$, $\phi(x) = \max\{x, \alpha x\} = (1-\alpha)\sigma\left(\left(\right) x\right) + \alpha x$.

We have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k}}\left[\phi(\boldsymbol{a}_i^\top \boldsymbol{z})\phi(\boldsymbol{a}_j^\top \boldsymbol{z})\right] \\
=&(1-\alpha)^2\,\mathbb{E}_{\boldsymbol{z}}\,\sigma\left(\left(\right)\boldsymbol{a}_i^\top \boldsymbol{z}\right)\sigma\left(\left(\right)\boldsymbol{a}_j^\top \boldsymbol{z}\right) + (1-\alpha)\alpha\,\mathbb{E}_{\boldsymbol{z}}\,\sigma\left(\left(\right)\boldsymbol{a}_i^\top \boldsymbol{z}\right)\boldsymbol{a}_j^\top \boldsymbol{z} \\
&+ (1-\alpha)\alpha\,\mathbb{E}_{\boldsymbol{z}}\,\boldsymbol{a}_i^\top \boldsymbol{z}\sigma\left(\left(\right)\boldsymbol{a}_j^\top \boldsymbol{z}\right) + \alpha^2\,\mathbb{E}_{\boldsymbol{z}}\,\boldsymbol{a}_i^\top \boldsymbol{z}\boldsymbol{a}_j^\top \boldsymbol{z} \\
=&(1-\alpha)^2 h(\boldsymbol{a}_i^\top \boldsymbol{a}_j) + \alpha \boldsymbol{a}_i^\top \boldsymbol{a}_j
\end{aligned}$$

Therefore for leaky ReLU $\tilde{g}_{jk}(z) = \frac{1}{2}((1-\alpha)^2(h(z) - h(z_{jk^*})) + \alpha(z - z^*_{jk}))^2$, and $\tilde{g}'_{jk}(z) = ((1-\alpha)^2(h(z)-h(z_{jk^*}))+\alpha(z-z^*_{jk}))((1-\alpha)^2 h'(z)+\alpha)$. Now with $\alpha > 0$, $(1-\alpha)^2 h'(z)+\alpha \geq \alpha$ for all z and $\tilde{g}_{jk}(z) = 0 \Leftrightarrow z = z^*_{jk}$.

To sum up, for odd activations and leaky ReLU, since each $\tilde{g}_{jk}(z)$ only has stationary point of $z = z^*_{jk}$, the stationary point Z of $\tilde{g}(Z) = \sum_{jk} \tilde{g}_{jk}$ also satisfy $Z = Z^* = A^*(A^*)^\top$.

□

Proof of Theorem 4.5.3. Instead of directly looking at the second-order stationary point of Problem 1, we look at the following problem on its reparametrized version:

Problem 2.

$$\begin{aligned}
\min_Z \quad & \left\{\tilde{g}(Z) = \frac{1}{2}\left\|\sum_{i=0}^{\infty}\sigma_i^2\left((Z^*)^{\circ i} - Z^{\circ i}\right)\right\|_F^2\right\} \\
\textit{s.t.} \quad & z_{ii} = 1, \forall i. \\
& Z \succeq 0.
\end{aligned}$$

Here $Z^* = A^*(A^*)^\top$ and satisfies $z^*_{ii} = 1, \forall i$.

Compared to function g in the original problem 1, it satisfies that $\tilde{g}(AA^\top) \equiv g(A)$.

A matrix Z satisfies the first-order stationary point for Problem 2 if there exists a vector σ such that:

$$\begin{cases} z_{ii} = 1, \\ Z \succeq 0, \\ S \succeq 0, \\ SZ = 0, \\ S = \nabla_Z g(Z) - \text{diag}(\sigma). \end{cases}$$

Therefore for a stationary point Z, since $Z^* = A^*(A^*)^\top \succeq 0$, and $S \succeq 0$, we have $\langle S, Z^* - Z\rangle = \langle S, Z^*\rangle \geq 0$. Meanwhile,

$$\begin{aligned} &\langle Z^* - Z, S\rangle \\ =&\langle Z^* - Z, \nabla_Z f(Z) - \text{diag}(\sigma)\rangle \\ =&\langle Z^* - Z, \nabla_Z f(Z)\rangle && (\text{diag}(Z^* - Z) = 0) \\ =&\sum_{i,j}(z^*_{ij} - z_{ij})g'_{ij}(z_{ij}) \\ =&\sum_{i,j}(z_{ij} - z^*_{ij})P(z_{ij})(z^*_{ij} - z_{ij}) \\ &&(\text{Refer to proof of Lemma 4.5.2 for the value of } g') \\ =&-\sum_{ij}(z_{ij} - z^*_{ij})^2 P(z_{ij}) \\ \leq&0 && (P \text{ is always positive}) \end{aligned}$$

Therefore $\langle S, Z^* - Z\rangle = 0$, and this only happens when $Z = Z^*$.

Finally, from [75] we know that any first-order stationary point for Problem 2 is a second-order stationary point for our original problem 1 [2]. Therefore we conclude that all second-order

[2]Throughout the analysis for low rank optimization in [75], they require function $\tilde{g}(Z)$ to be convex. However, by

stationary point for Problem 1 are global minimum A: $AA^\top = A^*(A^*)^\top$. □

C.2.4 Landscape Analysis for Non-unit Generating Vectors

In the previous argument, we simply assume that the norm of each generating vectors $\boldsymbol{a}_i$ to be 1. This practice simplifies the computation but is not practical. Since we are able to estimate $\|\boldsymbol{a}_i\|$ for all i first, we could analyze the landscape of our loss function for general matrix A.

The main tool is to use the multiplication theorem of Hermite functions:

$$h_n^\alpha(x) := h_n(\alpha x) = \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \alpha^{n-2i}(\alpha^2-1)^i \binom{n}{2i} \frac{(2i)!}{i!} 2^{-i} h_{n-2i}(x).$$

For the ease of notation, we denote the coefficient as $\eta_\alpha^{n,i} := \alpha^{n-2i}(\alpha^2-1)^i \binom{n}{2i} \frac{(2i)!}{i!} 2^{-i}$. We extend the calculations for Hermite inner product for non-standard distributions.

Lemma C.2.2. *Let (x, y) be normal variables that follow joint distribution $\mathcal{N}(0, [[\alpha^2, \alpha\beta\rho]; [\alpha\beta\rho, \beta^2]])$. Then,*

$$\mathbb{E}[h_m(x)h_n(y)] = \begin{cases} \sum_{i=0}^{\lfloor \frac{l}{2} \rfloor} \eta_\alpha^{l,i}\eta_\beta^{l,i}\rho^{l-2i} & \text{if } m \equiv n \ (\text{mod } 2) \\ 0 & \text{o.w.} \end{cases} \tag{C.2}$$

Here $l = \min\{m, n\}$.

Proof. Denote the normalized variables $\hat{x} = x/\alpha$, $\hat{y} = y/\beta$. Let $l = \min\{m, n\}$.

$$\begin{aligned} &\mathbb{E}[h_m(x)h_n(y)] \\ =\ &\mathbb{E}[h_m^\alpha(\hat{x})h_n^\beta(\hat{y})] \end{aligned}$$

carefully scrutinizing the proof, one could see that this condition is not required in building the connection of first-order and second-order stationary points of $g(A)$ and $\tilde{g}(Z)$. For more cautious readers, we also show a relaxed version in the next section, where the equivalence of SOSP of g and FOSP of $\tilde{g}$ is a special case of it.

$$
\begin{aligned}
&= \sum_{i=0}^{\lfloor \frac{m}{2} \rfloor} \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} \eta_\alpha^{m,i} \eta_\beta^{n,j} \, \mathbb{E}[h_{m-2i}(\hat{x}) h_{n-2j}(\hat{y})] \\
&= \sum_{i=0}^{\lfloor \frac{m}{2} \rfloor} \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} \eta_\alpha^{m,i} \eta_\beta^{n,j} \delta_{(m-2i),(n-2j)} \rho^{n-2j} \qquad \text{(Lemma C.2.2)} \\
&= \begin{cases} \sum_{i=0}^{\lfloor \frac{l}{2} \rfloor} \eta_\alpha^{l,i} \eta_\beta^{l,i} \rho^{l-2i} & \text{if } m \equiv n \pmod 2 \\ 0 & \text{o.w.} \end{cases}.
\end{aligned}
$$

□

Now the population risk becomes

$$
\begin{aligned}
g(A) =& \frac{1}{2} \left\| \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}} \left[\boldsymbol{x}\boldsymbol{x}^\top \right] - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \left[\phi(A\boldsymbol{z}) \phi(A\boldsymbol{z})^\top \right] \right\|^2 \\
=& \frac{1}{2} \sum_{i,j \in [d]} \left(\mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k_0 \times k_0})} \phi((\boldsymbol{a}_i^*)^\top \boldsymbol{z}) \phi((\boldsymbol{a}_j^*)^\top \boldsymbol{z}) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} \phi(\boldsymbol{a}_i^\top \boldsymbol{z}) \phi(\boldsymbol{a}_j^\top \boldsymbol{z}) \right)^2 \\
\equiv& \frac{1}{2} \sum_{i,j} \tilde{g}_{ij}(z_{ij}).
\end{aligned}
$$

To simplify the notation, for a specific i, j pair, we write $\hat{x} = \boldsymbol{a}_i^\top \boldsymbol{z}/\alpha$, $\alpha = \|\boldsymbol{a}_i\|$ and $\hat{y} = \boldsymbol{a}_j^\top \boldsymbol{z}/\beta$, where $\beta = \|\boldsymbol{a}_j\|$. Namely we have $(\hat{x}, \hat{y}) \sim \mathcal{N}(0, [[1, \rho]; [\rho, 1]])$, where $\rho = \cos\langle \boldsymbol{a}_i, \boldsymbol{a}_j \rangle$. Again, recall $\phi(\alpha\hat{x}) = \sum_{k \text{ odd}} \sigma_i h_i(\alpha\hat{x}) = \sum_{k \text{ odd}} \sigma_i h_i^\alpha(\hat{x})$.

$$
\begin{aligned}
& \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0, I_{k \times k})} [\phi(\alpha\hat{x}) \phi(\beta\hat{y})] \\
=& \, \mathbb{E} \left[\sum_{m \text{ odd}} \sigma_m h_m^\alpha(\hat{x}) \sum_{n \text{ odd}} \sigma_n h_n^\beta(\hat{y}) \right] \\
=& \sum_{m,n \text{ odd}} \sigma_m \sigma_n \, \mathbb{E}_S [h_m^\alpha(\hat{x}) h_n^\beta(\hat{y})] \\
=& \sum_{m \text{ odd}} \sigma_m \sum_{n \le m \text{ odd}} \sigma_n \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \eta_\alpha^{n,k} \eta_\beta^{n,k} \rho^{n-2k}
\end{aligned}
$$

Therefore we could write out explicitly the coefficient for each term ρ^k, k odd, as:

$$c_k = \sum_{n \geq k \text{ odd}} \sigma_n \eta_\alpha^{n, \frac{n-k}{2}} \eta_\beta^{n, \frac{n-k}{2}} (\sum_{m \geq n} \sigma_m).$$

We have $\tilde{g}_{ij}(z_{ij}) = (\sum_{k \text{ odd}} c_k z_{ij}^k - \sum_{k \text{ odd}} c_k (z_{ij}^*)^k)^2$.

Now suppose σ_i to have the same sign, and $\|\alpha_i\| \geq 1, \forall$ or $\|\alpha_i\| \leq 1, \forall i$, each coefficient $c_i \geq 0$. Therefore still the only stationary point for $g(Z)$ is Z^*.

C.3 Omitted Proofs for Sample Complexity

C.3.1 Omitted Proofs for Relation on Approximate Stationary Points

Proof of Lemma 4.6.7. We first review what we want to prove. For a matrix A that satisfies ϵ-approximate SOSP for Eqn. (4.4), we define $S_A = \nabla_Z \tilde{g}(AA^\top) - \sum_{i=1}^n \lambda_i X_i$. The conditions ensure that A, λ, S_A satisfy:

$$\begin{cases} \text{Tr}(A^\top X_i A) = y_i, & \\ \|S_A \tilde{\boldsymbol{a}}_i\|_2 \leq \epsilon \|\tilde{\boldsymbol{a}}_i\|_2, & \{\tilde{\boldsymbol{a}}_j\}_j \text{ span the column space of } A \\ \text{Tr}(B^\top D_A \nabla_A \mathcal{L}(A, \lambda)[B]) \geq -\epsilon \|B\|_F^2, & \forall B \text{ s.t. } \text{Tr}(B^\top X_i A) = 0. \end{cases} \tag{C.3}$$

We just want to show $Z := AA^\top, \sigma := \lambda$, and $S := S_A$ satisfies the conditions for ϵ-FOSP of Eqn. (4.5). Therefore, by going over the conditions, its easy to tell that all other conditions automatically apply and it remains to show $S_A \succeq -\epsilon I$.

By noting that $\nabla_A \mathcal{L}(A, \lambda) = 2 S_A A$, one has:

$$\begin{aligned} &\frac{1}{2} \text{Tr}(B^\top D_A \nabla_A \mathcal{L}(A, \lambda)[B]) \\ =&\text{Tr}(B^\top S_A B) + \text{Tr}(B^\top D_A \nabla_Z \tilde{g}(AA^\top)[B] A) - \sum_{i=1}^n D_A \lambda_i[B] \text{Tr}(B^\top X_i A) \end{aligned}$$

(from Lemma 5 of [75])

$$=\text{Tr}(B^\top S_A B) + \text{Tr}(AB^\top D_A \nabla_Z \tilde{g}(AA^\top)[B]) \tag{C.4}$$

(From Eqn. (C.3) we have $\text{Tr}(B^\top X_i A) = 0$)

Notice that $A \in \mathbb{R}^{d\times k}$ and we have chosen $k = d$ for simplicity. We first argue when A is rank-deficient, i.e. $\text{rank}(A) < k$. There exists some vector $\boldsymbol{v} \in \mathbb{R}^k$ such that $A\boldsymbol{v} = 0$. Now for any vector $\boldsymbol{b} \in \mathbb{R}^d$, let $B = \boldsymbol{b}\boldsymbol{v}^\top$. Therefore $AB^\top = A\boldsymbol{v}\boldsymbol{b}^\top = 0$. From (C.4) we further have:

$$\begin{aligned}
&\frac{1}{2}\text{Tr}(B^\top D_A \nabla_A \mathcal{L}(A,\lambda)[B]) \\
=&\text{Tr}(B^\top S_A B) + \text{Tr}(AB^\top D_A \nabla_Z \tilde{g}(AA^\top)[B]) \\
=&\text{Tr}(\boldsymbol{v}\boldsymbol{b}^\top S_A \boldsymbol{b}\boldsymbol{v}^\top) = \|\boldsymbol{v}\|^2 \boldsymbol{b}^\top S_A \boldsymbol{b} \\
\geq& -\epsilon/2\|B\|_F^2 && \text{(from (C.3))} \\
=& -\epsilon/2\|\boldsymbol{v}\|^2\|\boldsymbol{b}^\top\|^2
\end{aligned}$$

Therefore from the last three rows we have $\boldsymbol{b}^\top S_A \boldsymbol{b} \geq -\epsilon/2\|\boldsymbol{b}\|^2$ for any $\boldsymbol{b}$, i.e. $S_A \succeq -\epsilon/2 I_{d\times d}$. On the other hand, when A is full rank, the column space of A is the entire $\mathbb{R}^d$ vector space, and therefore $S_A \succeq -\epsilon I_{d\times d}$ directly follows from the second line of the ϵ-SOSP definition.

□

C.3.2 Detailed Calculations

Recall the population risk

$$g(A) \equiv \frac{1}{2}\left\|\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\left[\boldsymbol{x}\boldsymbol{x}^\top\right] - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\|_F^2.$$

Write the empirical risk on observations as:

$$g_n(A) \equiv \frac{1}{2}\left\|\frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\|_F^2.$$

Claim C.3.1.

$$\nabla g(A) - \nabla g_n(A) = 2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi'(A\boldsymbol{z}))(X - X_n)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right],$$

where $X = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}\boldsymbol{x}^\top]$, *and* $X_n = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top$.

Proof.

$$\begin{aligned}
&\nabla g(A) - \nabla g_n(A) = \nabla(g(A) - g_n(A)) \\
=&\frac{1}{2}\nabla\left\langle X - X_n, X + X_n - 2\,\mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\rangle \\
=&\nabla\left\langle X_n - X, \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\rangle
\end{aligned}$$

Now write $S(A) = \phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top$.

$$\begin{aligned}
&[S(A+\Delta A) - S(A)]_{ij} \\
=&\phi(\boldsymbol{a}_i^\top\boldsymbol{z} + \Delta\boldsymbol{a}_i^\top\boldsymbol{z})\phi(\boldsymbol{a}_j^\top\boldsymbol{z} + \Delta\boldsymbol{a}_j^\top\boldsymbol{z}) - \phi(\boldsymbol{a}_i^\top\boldsymbol{z})\phi(\boldsymbol{a}_j^\top\boldsymbol{z}) \\
=&\phi'(a_i^\top\boldsymbol{z})\Delta\boldsymbol{a}_i^\top\boldsymbol{z}\phi(\boldsymbol{a}_j^\top\boldsymbol{z}) + \phi'(a_j^\top\boldsymbol{z})\Delta\boldsymbol{a}_j^\top\boldsymbol{z}\phi(\boldsymbol{a}_i^\top\boldsymbol{z}) + \mathcal{O}(\|\Delta A\|^2)
\end{aligned}$$

Therefore

$$\begin{aligned}
&[S(A+\Delta A) - S(A)]_{i:} \\
=&\phi'(a_i^\top\boldsymbol{z})\Delta\boldsymbol{a}_i^\top\boldsymbol{z}\phi(A\boldsymbol{z})^\top + (\phi'(A\boldsymbol{z})\circ\Delta A\boldsymbol{z})^\top\phi(\boldsymbol{a}_i^\top\boldsymbol{z}) + \mathcal{O}(\|\Delta A\|^2)
\end{aligned}$$

Therefore

$$S(A+\Delta A) - S(A) = \mathrm{diag}(\phi'(A\boldsymbol{z}))\Delta A\boldsymbol{z}\phi(A\boldsymbol{z})^\top + \phi(A\boldsymbol{z})\boldsymbol{z}^\top\Delta A^\top\mathrm{diag}(\phi'(A\boldsymbol{z})). \tag{C.5}$$

And

$$g(A+\Delta A) - g_n(A+\Delta A) - (g(A) - g_n(A))$$

$$
\begin{aligned}
=&\langle X_n - X, \mathbb{E}_{\boldsymbol{z}}\left[S(A+\Delta A) - S(A)\right]\rangle \\
=&\,\mathbb{E}_{\boldsymbol{z}}\langle X_n - X, \mathrm{diag}(\phi'(A\boldsymbol{z}))\Delta A\boldsymbol{z}\phi(A\boldsymbol{z})^\top + \phi(A\boldsymbol{z})\boldsymbol{z}^\top\Delta A^\top \mathrm{diag}(\phi'(A\boldsymbol{z}))\rangle \\
=&2\,\mathbb{E}_{\boldsymbol{z}}\langle \mathrm{diag}(\phi'(A\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top, \Delta A\rangle.
\end{aligned}
$$

Finally we have $\nabla g(A) - \nabla g_n(A) = 2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathrm{diag}(\phi'(A\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right]$. □

Claim C.3.2. *For arbitrary matrix B, the directional derivative of $\nabla g(A) - \nabla g_n(A)$ with direction B is:*

$$
\begin{aligned}
& D_A\nabla g(A)[B] - D_A\nabla g_n(A)[B] \\
= \;& 2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi'(A\boldsymbol{z}))(X_n - X)\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z})\boldsymbol{z}^\top\right] \\
& +2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathit{diag}(\phi''(A\boldsymbol{z}) \circ (B\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right]
\end{aligned}
$$

Proof.

$$
\begin{aligned}
& g(A+tB) \\
=&2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathrm{diag}(\phi'(A\boldsymbol{z}+tB\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z}+tB\boldsymbol{z})\boldsymbol{z}^\top\right] \\
=&2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathrm{diag}(\phi'(A\boldsymbol{z}) + t(B\boldsymbol{z}) \circ \phi''(A\boldsymbol{z}))(X_n - X)(\phi(A\boldsymbol{z}) + t\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z}))\boldsymbol{z}^\top\right] + \mathcal{O}(t^2)
\end{aligned}
$$

Therefore

$$
\begin{aligned}
& \lim_{t\to 0}\frac{g(A+tB) - g(A)}{t} \\
=&2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathrm{diag}(\phi'(A\boldsymbol{z}))(X_n - X)\phi'(A\boldsymbol{z}) \circ (B^\top\boldsymbol{z})\boldsymbol{z}^\top\right] \\
& + 2\,\mathbb{E}_{\boldsymbol{z}}\left[\mathrm{diag}(\phi''(A\boldsymbol{z}) \circ (B\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right]
\end{aligned}
$$

□

C.3.3 Omitted Proofs for Observation Sample Complexity

Proof of Lemma 4.6.2. For each $\boldsymbol{x}_i = \phi(A\boldsymbol{z}_i), \boldsymbol{z}_i \sim \mathcal{N}(0, I_{k\times k})$. Each coordinate $|x_{i,j}| = |\phi(\boldsymbol{a}_j^\top \boldsymbol{z}_i)| \leq |\boldsymbol{a}_j^\top \boldsymbol{z}_i|$ since ϕ is 1-Lipschitz. [3]. Without loss of generality we assumed $\|\boldsymbol{a}_j\| = 1, \forall j$, therefore $\boldsymbol{a}_j^\top \boldsymbol{z} \sim \mathcal{N}(0, I_{k\times k})$. For all $i \in [n], j \in [d]$ $|x_{i,j}| \leq \log(nd/\delta)$ with probability $1-\delta$.

Then by matrix concentration inequality ([159] Corollary 5.52), we have with probability $1-\delta$: $(1-\epsilon)X \preceq X_n \preceq (1+\epsilon)X$ if $n \geq \Omega(d/\epsilon^2 \log^2(nd/\delta))$. Therefore set $n = \tilde{\Theta}(d/\epsilon^2 \log^2(1/\delta))$ will suffice. □

Proof of Lemma 4.6.3.

$$\begin{aligned} X_{ij} &= \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})} \phi(\boldsymbol{a}_i^\top \boldsymbol{z})\phi(\boldsymbol{a}_j^\top \boldsymbol{z}) \\ &= \begin{cases} 0 & i \neq j \\ \mathbb{E}[\phi^2(\boldsymbol{a}_i^\top \boldsymbol{z})] \leq \frac{2}{\pi} & i = j \end{cases} \end{aligned}$$

Therefore $\|X\|_2 \leq \frac{2}{\pi}$. Together with Lemma 4.6.2, $\|X - X_n\| \leq \epsilon\frac{2}{\pi}$ w.p $1-\delta$. Recall

$$\nabla g(A) - \nabla g_n(A) = 2\,\mathbb{E}_{\boldsymbol{z}} \left[\text{diag}(\phi'(A\boldsymbol{z}))(X - X_n)\phi(A\boldsymbol{z})\boldsymbol{z}^\top\right] := 2\,\mathbb{E}_{\boldsymbol{z}}\, G(\boldsymbol{z}),$$

where $G(\boldsymbol{z})$ is defined as $\text{diag}(\phi'(A\boldsymbol{z}))(X-X_n)\phi(A\boldsymbol{z})\boldsymbol{z}^\top$. We have $\|G(z)\| \leq \|A\|\|\boldsymbol{z}\|^2\|X-X_n\|$.

$$\begin{aligned} \|\nabla g(A) - \nabla g_n(A)\|_2 &= 2\|\,\mathbb{E}_{\boldsymbol{z}}[G(\boldsymbol{z})]\| \\ &\leq 2\,\mathbb{E}_{\boldsymbol{z}}\,\|G(\boldsymbol{z})\| \\ &\leq 2\,\mathbb{E}_{\boldsymbol{z}}\,\|A\|\|\boldsymbol{z}\|^2\|X - X_n\| \\ &\leq 2\|A\|\epsilon\frac{2}{\pi}\,\mathbb{E}_{\boldsymbol{z}}\,\|\boldsymbol{z}\|^2 \end{aligned}$$

[3] For simplicity, we analyze as if $\phi(0) = 0$ w.o.l.g. throughout this section, since the bias term is canceled out in the observation side with $\phi(A^*\boldsymbol{z})$ and the learning side with $\phi(A\boldsymbol{z})$.

$$= \quad 2\|A\|\epsilon d\frac{2}{\pi}$$

For the directional derivative, we make the concentration bound in a similar way. Denote

$$D(\boldsymbol{z}) = \text{diag}(\phi'(A\boldsymbol{z}))(X_n - X)\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z})\boldsymbol{z}^\top + \text{diag}(\phi''(A\boldsymbol{z}) \circ (B\boldsymbol{z}))(X_n - X)\phi(A\boldsymbol{z})\boldsymbol{z}^\top.$$

$$\|D(\boldsymbol{z})\| \leq \|X_n - X\|_2\|B\|\|\boldsymbol{z}\|^2(1 + \|\boldsymbol{z}\|\|A\|).$$

Therefore $\|D_A\nabla g(A)[B] - D_A\nabla g_n(A)[B]\| \leq \mathcal{O}(\epsilon d^{3/2}\|A\|\|B\|)$ with probability $1 - \delta$. □

C.3.4 Omitted Proofs on Bounding Mini-Batch Size

Recall

$$\tilde{g}_{m,n}(A) \equiv \frac{1}{2}\left\|\frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top - \frac{1}{m}\sum_{j=1}^m \phi(A\boldsymbol{z}_j)\phi(A\boldsymbol{z}_j)^\top\right\|_F^2.$$

Write $S_j(A) \equiv \phi(A\boldsymbol{z}_j)\phi(A\boldsymbol{z}_j)^\top$. Then we have

$$\begin{aligned}\tilde{g}_{m,n}(A) =& \frac{1}{2}\left\langle X_n - \frac{1}{n}\sum_{j=1}^m S_j(A), X_n - \frac{1}{m}\sum_{j=1}^m S_j(A)\right\rangle \\ =& \frac{1}{2m^2}\sum_{i,j}\langle S_i(A), S_j(A)\rangle - \frac{1}{n}\sum_{j=1}^m\langle S_j(A), X_n\rangle + \frac{1}{2}\|X_n\|_F^2\end{aligned}$$

On the other hand, our target function is:

$$\begin{aligned}g_n(A) \equiv& \frac{1}{2}\left\|\frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i\boldsymbol{x}_i^\top - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{N}(0,I_{k\times k})}\left[\phi(A\boldsymbol{z})\phi(A\boldsymbol{z})^\top\right]\right\|_F^2 \\ =& \frac{1}{2}\|\mathbb{E}_S[S]\|_F^2 - \langle\mathbb{E}_S[S], X_n\rangle + \frac{1}{2}\|X_n\|_F^2\end{aligned}$$

Therefore $\mathbb{E}_S\,\tilde{g}_{m,n}(A) - g_n(A) = \frac{1}{2m}(\mathbb{E}_S\,\|S(A)\|_F^2 - \|\,\mathbb{E}_S\,S(A)\|_F^2)$.

Claim C.3.3.

$$\nabla \mathbb{E}_S \tilde{g}_{m,n}(A) - \nabla g_n(A) = \frac{2}{m} \mathbb{E}_{\boldsymbol{z}} \left[\textit{diag}(\phi'(A\boldsymbol{z})) S(A) \phi(A\boldsymbol{z}) \boldsymbol{z}^\top - \textit{diag}(\phi'(A\boldsymbol{z})) \, \mathbb{E}_S[S(A)] \phi(A\boldsymbol{z}) \boldsymbol{z}^\top \right].$$

Proof.

$$\begin{aligned}
&\langle \nabla \mathbb{E}_S \tilde{g}_{m,n} - \nabla g_n, \Delta A \rangle \\
=& \mathbb{E}_S \tilde{g}_{m,n}(A + \Delta A) + g_n(A + \Delta A) - (\mathbb{E}_S \tilde{g}_{m,n}(A) + g_n(A)) + \mathcal{O}(\|\Delta A\|^2) \\
=& \frac{1}{2m} \left(\mathbb{E}_S \|S(A + \Delta A)\|_F^2 - \mathbb{E}_S \|S(A)\|_F^2 - \| \mathbb{E}_S S(A + \Delta A)\|_F^2 + \| \mathbb{E}_S S(A)\|_F^2 \right) + \mathcal{O}(\|\Delta A\|^2) \\
=& \frac{1}{m} \left(\mathbb{E}_S \langle S(A), S(A + \Delta A) - S(A) \rangle - \langle \mathbb{E}_S[S(A)], E_S[S(A + \Delta A) - S(A)] \right) + \mathcal{O}(\|\Delta A\|^2) \\
=& \frac{1}{m} \left(\langle \mathbb{E}_{\boldsymbol{z}} \langle S(A), \operatorname{diag}(\phi'(A\boldsymbol{z})) \Delta A \boldsymbol{z} \phi(A\boldsymbol{z})^\top \rangle - \langle \mathbb{E}_S[S(A)], \mathbb{E}_{\boldsymbol{z}} \operatorname{diag}(\phi'(A\boldsymbol{z})) \Delta A \boldsymbol{z} \phi(A\boldsymbol{z})^\top \rangle \right) \\
&+ \mathcal{O}(\|\Delta A\|^2) \qquad \text{(from Eqn. (C.5) and symmetry of } S\text{)} \\
=& \langle \frac{2}{m} \mathbb{E}_{\boldsymbol{z}} \left[\operatorname{diag}(\phi'(A\boldsymbol{z})) S(A) \phi(A\boldsymbol{z}) \boldsymbol{z}^\top - \operatorname{diag}(\phi'(A\boldsymbol{z})) \, \mathbb{E}_S[S(A)] \phi(A\boldsymbol{z}) \boldsymbol{z}^\top \right], \Delta A \rangle + \mathcal{O}(\|\Delta A\|^2)
\end{aligned}$$

□

Similarly to the derivation in the previous subsection, we again derive the bias in the directional derivative:

Claim C.3.4. *For arbitrary matrix direction B,*

$$\begin{aligned}
&D_A \nabla \mathbb{E}_S \tilde{g}_{m,n}(A)[B] - D_A \nabla g_n(A)[B] \\
=& \frac{2}{m} \mathbb{E}_{\boldsymbol{z}} \left[\textit{diag}(\phi''(A\boldsymbol{z}) \circ (B\boldsymbol{z})) (S(A) - \mathbb{E}_S S(A)) \phi(A\boldsymbol{z}) \boldsymbol{z}^\top \right. \\
&+ \textit{diag}(\phi'(A\boldsymbol{z})) \left((\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z})) \phi(A\boldsymbol{z})^\top - \mathbb{E}_{\boldsymbol{z}}[(\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z})) \phi(A\boldsymbol{z})^\top] \right) \phi(A\boldsymbol{z}) \boldsymbol{z}^\top \\
&+ \textit{diag}(\phi'(A\boldsymbol{z})) \left(\phi(A\boldsymbol{z}) (\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z}))^\top - \mathbb{E}_{\boldsymbol{z}}[\phi(A\boldsymbol{z}) (\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z}))^\top] \right) \phi(A\boldsymbol{z}) \boldsymbol{z}^\top \\
&\left. + \textit{diag}(\phi'(A\boldsymbol{z})) (S(A) - \mathbb{E}_S S(A)) (\phi'(A\boldsymbol{z}) \circ (B\boldsymbol{z})) \boldsymbol{z}^\top \right]
\end{aligned}$$

C.3.5 Omitted Proof of the Main Theorem

Proof of Lemma 4.6.8. On one hand, suppose Z is an ϵ-FOSP property of $\tilde{g}$ in (4.5) along with the matrix S and vector σ, we have:

$$\begin{aligned}
&\langle \nabla \tilde{g}(Z), Z - Z^* \rangle \\
=&\langle S, Z - Z^* \rangle \\
&\qquad\qquad \text{(since } Z - Z^* \text{ has 0 diagonal entries)} \\
\leq&\|P_T(S)\|_2 \|P_{T^\circ}(Z - Z^*)\|_F \\
&\qquad\qquad (T \text{ is the tangent cone of PSD matrices at } Z) \\
\leq&\|P_T(S)\|_2 \|Z - Z^*\|_F \\
=&\max_j \{\tilde{\boldsymbol{a}}_j^\top S \tilde{\boldsymbol{a}}_j\} \|Z - Z^*\|_F \\
&\qquad\qquad (\tilde{\boldsymbol{a}}_j \text{ is the basis of the column space of } Z\text{)} \\
\leq&\epsilon \|Z - Z^*\|_F \qquad\qquad \text{(C.6)} \\
&\qquad\qquad \text{(from the definition of } \epsilon\text{-FOSP)}
\end{aligned}$$

On the other hand, from the definition of $\tilde{g}$, we have:

$$\begin{aligned}
&\langle Z - Z^*, \nabla \tilde{g}(Z) \rangle \\
=&\sum_{ij} (z_{ij} - z_{ij}^*) \tilde{g}'_{ij}(z_{ij}) \\
=&\sum_{ij} (z_{ij} - z_{ij}^*)^2 \sum_{k \text{ odd}} \sigma_k^2 P_k(z_{ij}) \sum_{k \text{ odd}} \sigma_k^2 k z_{ij}^{k-1} \\
\geq&\|Z - Z^*\|_F^2 \sigma_1^4 \qquad\qquad \text{(C.7)}
\end{aligned}$$

Here polynomial $P_k(z_{ij}) \equiv (z_{ij}^k - (z_{ij}^*)^k)/(z - z^*)$ is always positive for $z \neq z^*$ and k to be odd.

Therefore by comparing (C.6) and (C.7) we have $\epsilon\|Z - Z^*\|_F \geq \|Z - Z^*\|_F^2\sigma_1^4$, i.e. $\|Z - Z^*\|_F \leq \mathcal{O}(\epsilon)$. □

Proof of Theorem 4.6.9. From Theorem 31 from [57], we know for small enough learning rate η, and arbitrary small ϵ, there exists large enough T, such that Algorithm 3 generates an output $A^{(T)}$ that is sufficiently close to the second order stationary point for f. Or formally we have,

$$\begin{cases} \text{Tr}((A^{(T)})^\top X_i A^{(T)}) = y_i, \\ \|(\nabla_A f(A^{(T)}) - \sum_{i=1} \lambda_i X_i A^{(T)})_{:,j}\|_2 \leq \epsilon \min \|A_{j,:}\|_2, & \forall j \in [k] \\ \text{Tr}(B^\top D_A \nabla_A \mathcal{L}_f(A^{(T)}, \lambda)[B]) \geq -\epsilon\|B\|_2^2, & \forall B, s.t. \text{Tr}(B^\top X_i A) = 0 \end{cases}$$

$\mathcal{L}_f(A, \lambda) = f(A) - \sum_{i=1}^d \lambda_i(\text{Tr}(A^\top X_i A) - y_i)$. Let $\{\tilde{\boldsymbol{a}}_i = A^{(T)}\boldsymbol{r}_i\}_i^k$ to form the basis of the column vector space of $A^{(T)}$. Then the second line is a sufficient condition for the following: $\|\tilde{\boldsymbol{a}}_j^\top(\nabla_A f(A^{(T)}) - \sum_{i=1} \lambda_i X_i A^{(T)})\boldsymbol{r}_j\|_2 \leq \epsilon, \forall j \in [k]$.

Now with the concentration bound from Lemma 4.6.4, suppose our batch size $m \geq \mathcal{O}(d^5/\epsilon)$, we have $\|\nabla_A g_n(A^{(T)}) - \nabla_A f(A^{(T)})\|_2 \leq \epsilon$, and $\|D_A \nabla_A g_n(A^{(T)})[B] - D_A \nabla_A f(A^{(T)})[B]\|_2 \leq \epsilon\|B\|_2$ for arbitrary B. Therefore again we get:

$$\begin{cases} \text{Tr}((A^{(T)})^\top X_i A^{(T)}) = y_i \\ \|\tilde{\boldsymbol{a}}_j^\top(\nabla_A g_n(A^{(T)}) - \sum_{i=1} \lambda_i X_i A^{(T)})\boldsymbol{r}_j\|_2 \leq 2\epsilon, & \forall j \in [k] \\ \text{Tr}(B^\top D_A \nabla_A \mathcal{L}_{g_m}(A^{(T)}, \lambda)[B]) \geq -2\epsilon\|B\|_2^2, & \forall B, s.t. \text{Tr}(B^\top X_i A) = 0 \end{cases}$$

Next we turn to the concentration bound from Lemma 4.6.3. Suppose we have when the sample size $n \geq \mathcal{O}(d^5/\epsilon^2 \log^2(1/\delta))$, $\|D_A \nabla_A g(A)[B] - D_A \nabla_A g_n(A)[B]\|_2 \leq \mathcal{O}(\epsilon\|B\|_2)$, and $\|\nabla g(A) - \nabla g_n(A)\|_2 \leq \mathcal{O}(\epsilon)$ with probability $1 - \delta$. Therefore similarly we get $A^{(T)}$ is an $\mathcal{O}(\epsilon)$-SOSP for $g(A) = \frac{1}{2}\left\|\sum_{i=0}^\infty \sigma_i^2 \left((A^*(A^*)^\top)^{\circ i} - (AA^\top)^{\circ i}\right)\right\|_F^2$.

Now with Lemma 4.6.7 that connects the approximate stationary points, we have $Z := A^{(T)}(A^{(T)})^\top$ is also an ϵ-FOSP of $\tilde{g}(Z) = \frac{1}{2}\left\|\sum_{i=0}^\infty \sigma_i^2 \left((Z^*)^{\circ i} - Z^{\circ i}\right)\right\|_F^2$.

Finally with Lemma 4.6.8, we get $\|Z - Z^*\|_F \leq \mathcal{O}(\epsilon)$.

□

Bibliography

[1] Jacob D. Abernethy, Kevin A. Lai, and Andre Wibisono. Last-iterate convergence rates for min-max optimization. *CoRR*, abs/1906.02027, 2019.

[2] Pierre-Antoine Absil, Robert E. Mahony, and B. Andrews. Convergence of the iterates of descent methods for analytic cost functions. *SIAM Journal on Optimization*, 16(2):531–547, 2005.

[3] Ilan Adler. The equivalence of linear programs and zero-sum games. In *International Journal of Game Theory*, pages 165–177, 2013.

[4] Leonard Adolphs, Hadi Daneshmand, Aurelien Lucchi, and Thomas Hofmann. Local saddle point optimization: A curvature exploitation approach. *arXiv preprint arXiv:1805.05751*, 2018.

[5] Zeyuan Allen-Zhu, Elad Hazan, Wei Hu, and Yuanzhi Li. Linear convergence of a Frank-Wolfe type algorithm over trace-norm balls. In *Advances in Neural Information Processing Systems*, 2017.

[6] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 2008.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223, 2017.

[8] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 214–223, 2017.

[9] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 224–232. JMLR. org, 2017.

[10] Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. Simple, efficient, and neural algorithms for sparse coding. In *28th Conference on Learning Theory*, pages 113–149, 2015.

[11] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[12] Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do GANs learn the distribution? some theory and empirics. 2018.

[13] Yu Bai, Tengyu Ma, and Andrej Risteski. Approximability of discriminators implies diversity in GANs. *arXiv preprint arXiv:1806.10586*, 2018.

[14] James P. Bailey and Georgios Piliouras. Multiplicative weights update in zero-sum games. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 321–338, 2018.

[15] Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

[16] David Blackwell. An analog of the minimax theorem for vector payoffs. In *Pacific J. Math.*, pages 1–8, 1956.

[17] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 537–546. JMLR. org, 2017.

[18] G.W Brown. Iterative solutions of games by fictitious play. In *Activity Analysis of Production and Allocation*, 1951.

[19] Christoph Buchheim and Jannis Kurtz. Min-max-min robustness: a new approach to combinatorial optimization under uncertainty based on multiple solutions. *Electronic Notes in Discrete Mathematics*, 52:45–52, 2016.

[20] Emmanuel J Candes, Xiaodong Li, and Mahdi Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. *Information Theory, IEEE Transactions on*, 61(4):1985–2007, 2015.

[21] Nikolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[22] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2011.

[23] Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. Training and testing low-degree polynomial data mappings via linear SVM. *Journal of Machine Learning Research*, 2010.

[24] Erick Chastain, Adi Livnat, Christos Papadimitriou, and Umesh Vazirani. Algorithms, games, and evolution. *Proceedings of the National Academy of Sciences (PNAS)*, 111(29):10620–10623, 2014.

[25] George HG Chen and R Tyrrell Rockafellar. Convergence rates in forward–backward splitting. *SIAM Journal on Optimization*, 7(2):421–444, 1997.

[26] JiaZhou Chen, Qi Lei, YongWei Miao, and QunSheng Peng. Vectorization of line drawing image based on junction analysis. *Science China Information Sciences*, 58(7):1–14, 2015.

[27] Jiazhou Chen, Qi Lei, Fan Zhong, and Qunsheng Peng. Interactive tensor field design based on line singularities. In *2013 International Conference on Computer-Aided Design and Computer Graphics*, pages 353–360. IEEE, 2013.

[28] Minhao Cheng, Qi Lei, Pin-Yu Chen, Inderjit Dhillon, and Cho-Jui Hsieh. Cat: Customized adversarial training for improved robustness. *arXiv preprint arXiv:2002.06789*, 2020.

[29] Ashish Cherukuri, Bahman Gharesifard, and Jorge Cortes. Saddle-point dynamics: conditions for asymptotic stability of saddle points. *SIAM Journal on Control and Optimization*, 55(1):486–511, 2017.

[30] Ashish Cherukuri, Bahman Gharesifard, and Jorge Cortés. Saddle-point dynamics: Conditions for asymptotic stability of saddle points. *SIAM J. Control and Optimization*, 55(1):486–511, 2017.

[31] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. *arXiv preprint arXiv:1412.0233*, 2014.

[32] Andrew R Conn, Nicholas IM Gould, and Ph L Toint. *Trust region methods*, volume 1. Siam, 2000.

[33] Bo Dai, Albert Shaw, Lihong Li, Lin Xiao, Niao He, Zhen Liu, Jianshu Chen, and Le Song. Sbeed: Convergent reinforcement learning with nonlinear function approximation. *arXiv preprint arXiv:1712.10285*, 2017.

[34] Eric Van Damme. *Stability and perfection of Nash equilibria*. Springer, 1991.

[35] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261, 2016.

[36] John M Danskin. *The theory of max-min and its application to weapons allocation problems*, volume 5. Springer Science & Business Media, 2012.

[37] George B Dantzig. A proof of the equivalence of the programming problem and the game problem. *Activity analysis of production and allocation*, (13):330–338, 1951.

[38] Constantinos Daskalakis, Themis Gouleakis, Chistos Tzamos, and Manolis Zampetakis. Efficient statistics, in high dimensions, from truncated samples. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 639–649. IEEE, 2018.

[39] Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. Training gans with optimism. *arXiv preprint arXiv:1711.00141*, 2017.

[40] Constantinos Daskalakis and Ioannis Panageas. Last-iterate convergence: Zero-sum games and constrained min-max optimization. *arXiv preprint arXiv:1807.04252*, 2018.

[41] Constantinos Daskalakis and Ioannis Panageas. The limit points of (optimistic) gradient descent in min-max optimization. In *Advances in Neural Information Processing Systems*, pages 9236–9246, 2018.

[42] Konstantinos Daskalakis. *The complexity of Nash equilibria*. 2008.

[43] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

[44] Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Nearest neighbor based greedy coordinate descent. In *Advances in Neural Information Processing Systems*, 2011.

[45] Ding-Zhu Du and Panos M Pardalos. *Minimax and applications*. Springer Science & Business Media, 2013.

[46] Simon S Du, Jianshu Chen, Lihong Li, Lin Xiao, and Dengyong Zhou. Stochastic variance reduction methods for policy evaluation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1049–1058. JMLR. org, 2017.

[47] Simon S Du and Wei Hu. Linear convergence of the primal-dual gradient method for convex-concave saddle point problems without strong convexity. *arXiv preprint arXiv:1802.01504*, 2018.

[48] Simon S Du, Wei Hu, Sham M Kakade, Jason D Lee, and Qi Lei. Few-shot learning via learning the representation, provably. *arXiv preprint arXiv:2002.09434*, 2020.

[49] Miroslav Dudik, Zaid Harchaoui, and Jérôme Malick. Lifted coordinate descent for learning with trace-norm regularization. In *Artificial Intelligence and Statistics*, 2012.

[50] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[51] Harrison Edwards and Amos Storkey. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*, 2015.

[52] Soheil Feizi, Farzan Farnia, Tony Ginart, and David Tse. Understanding GANs: the LQG setting. *arXiv preprint arXiv:1710.10793*, 2017.

[53] Tanner Fiez, Benjamin Chasnov, and Lillian J Ratliff. Convergence of learning dynamics in stackelberg games. *arXiv preprint arXiv:1906.01217*, 2019.

[54] Oded Galor. *Discrete Dynamical Systems*. Springer, 2007.

[55] Ruiqi Gao, Tianle Cai, Haochuan Li, Cho-Jui Hsieh, Liwei Wang, and Jason D Lee. Convergence of adversarial training in overparametrized neural networks. In *Advances in Neural Information Processing Systems*, pages 13009–13020, 2019.

[56] Dan Garber and Elad Hazan. Faster rates for the Frank-Wolfe method over strongly-convex sets. In *32nd International Conference on Machine Learning, ICML 2015*, 2015.

[57] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. *arXiv preprint arXiv:1503.02101*, 2015.

[58] Rong Ge, Jason D Lee, and Tengyu Ma. Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501*, 2017.

[59] Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. Negative momentum for improved game dynamics. In *the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[60] Donald Goldfarb, Garud Iyengar, and Chaoxu Zhou. Linear convergence of stochastic Frank Wolfe variants. *arXiv preprint arXiv:1703.07269*, 2017.

[61] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[62] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[63] Jacques Guélat and Patrice Marcotte. Some comments on Wolfe's 'away step'. *Mathematical Programming*, 1986.

[64] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[65] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[66] Elad Hazan and Haipeng Luo. Variance-reduced and projection-free stochastic optimization. In *International Conference on Machine Learning*, 2016.

[67] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[68] Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and Sellamanickam Sundararajan. A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th international conference on Machine learning*, pages 408–415. ACM, 2008.

[69] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[70] Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1627–1635. JMLR. org, 2017.

[71] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, 2013.

[72] Chi Jin, Praneeth Netrapalli, and Michael I Jordan. Minmax optimization: Stable limit points of gradient descent ascent are locally optimal. *arXiv preprint arXiv:1902.00618*, 2019.

[73] Chi Jin, Yuchen Zhang, Sivaraman Balakrishnan, Martin J. Wainwright, and Michael I. Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4116–4124, 2016.

[74] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

[75] Michel Journée, Francis Bach, P-A Absil, and Rodolphe Sepulchre. Low-rank optimization for semidefinite convex problems. *arXiv preprint arXiv:0807.4423*, 2008.

[76] Sham Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. On the duality of strong convexity and strong smoothness: Learning applications and matrix regularization.

[77] Sai Praneeth Karimireddy, Anastasia Koloskova, Sebastian U Stich, and Martin Jaggi. Efficient greedy coordinate descent for composite problems. *arXiv preprint arXiv:1810.06999*, 2018.

[78] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[79] John L. Kelley. *General Topology*. Springer, 1955.

[80] Thomas Kerdreux, Alexandre d'Aspremont, and Sebastian Pokutta. Restarting Frank-Wolfe. *International Conference on Machine Learning*, 2019.

[81] Raghunandan H Keshavan, Sewoong Oh, and Andrea Montanari. Matrix completion from a few entries. In *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*, pages 324–328. IEEE, 2009.

[82] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *Journal of the ACM (JACM)*, 2005.

[83] Robert Kleinberg, Georgios Piliouras, and Eva Tardos. Multiplicative updates outperform generic no-regret learning in congestion games. In *STOC*, pages 533–542. ACM, 2009.

[84] GM Korpelevich. The extragradient method for finding saddle points and other problems. *Matecon*, 12:747–756, 1976.

[85] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems*, 2015.

[86] Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMs. *arXiv preprint arXiv:1207.4747*, 2012.

[87] Guanghui Lan and Yi Zhou. Conditional gradient sliding for convex optimization. *SIAM Journal on Optimization*, 2016.

[88] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[89] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[90] Jason D. Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. First-order methods almost always avoid saddle points. *CoRR*, abs/1710.07406, 2017.

[91] Jason D. Lee, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 1246–1257, 2016.

[92] Qi Lei, Ajil Jalal, Inderjit S Dhillon, and Alexandros G Dimakis. Inverting deep generative models, one layer at a time. In *Advances in Neural Information Processing Systems*, pages 13910–13919, 2019.

[93] Qi Lei, Jason D Lee, Alexandros G Dimakis, and Constantinos Daskalakis. Sgd learns one-layer networks in wgans. *arXiv preprint arXiv:1910.07030*, 2019.

[94] Qi Lei, Sai Ganesh Nagarajan, Ioannis Panageas, and Xiao Wang. Last iterate convergence in no-regret learning: constrained min-max optimization for convex-concave landscapes. *arXiv preprint arXiv:2002.06768*, 2020.

[95] Qi Lei, Wei Sun, Roman Vaculin, and Jinfeng Yi. Method and system for time series representation learning via dynamic time warping, July 26 2018. US Patent App. 15/840,599.

[96] Qi Lei, Lingfei Wu, Pin-Yu Chen, Alexandros Dimakis, Inderjit Dhillon, and Michael Witbrock. Discrete adversarial attacks and submodular optimization with applications to text classification. *Systems and Machine Learning (SysML)*, 2019.

[97] Qi Lei, Ian EH Yen, Chao-yuan Wu, Inderjit S Dhillon, and Pradeep Ravikumar. Doubly greedy primal-dual coordinate descent for sparse empirical risk minimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

[98] Qi Lei, Jinfeng Yi, Roman Vaculin, Lingfei Wu, and Inderjit S Dhillon. Similarity preserving representation learning for time series clustering. *arXiv preprint arXiv:1702.03584*, 2017.

[99] Qi Lei, Kai Zhong, and Inderjit S Dhillon. Coordinate-wise power method. In *Advances in Neural Information Processing Systems*, 2016.

[100] Qi Lei, Jiacheng Zhuo, Constantine Caramanis, Inderjit S Dhillon, and Alexandros G Dimakis. Primal-dual block generalized frank-wolfe. In *Advances in Neural Information Processing Systems*, pages 13866–13875, 2019.

[101] Jerry Li, Aleksander Madry, John Peebles, and Ludwig Schmidt. Towards understanding the dynamics of generative adversarial networks.

[102] Tengyuan Liang. On how well generative adversarial networks learn densities: Nonparametric and parametric results. *arXiv preprint arXiv:1811.03179*, 2018.

[103] Tengyuan Liang and James Stokes. Interaction matters: A note on non-asymptotic local convergence of generative adversarial networks. In *the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[104] Qihang Lin, Mingrui Liu, Hassan Rafique, and Tianbao Yang. Solving weakly-convex-weakly-concave saddle-point problems as successive strongly monotone variational inequalities. *arXiv preprint arXiv:1810.10207*, 2018.

[105] Tianyi Lin, Chi Jin, and Michael I Jordan. On gradient descent ascent for nonconvex-concave minimax problems. *arXiv preprint arXiv:1906.00331*, 2019.

[106] Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Proximal gradient temporal difference learning algorithms. In *IJCAI*, pages 4195–4199, 2016.

[107] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. *arXiv preprint arXiv:1802.06309*, 2018.

[108] Tung Mai, Ioannis Panageas, Will Ratcliff, Vijay V. Vazirani, and Peter Yunker. Rock-paper-scissors, differential games and biological diversity. *CoRR*, abs/1710.11249, 2017.

[109] Eric V Mazumdar, Michael I Jordan, and S Shankar Sastry. On finding local nash equilibria (and only local nash equilibria) in zero-sum games. *arXiv preprint arXiv:1901.00838*, 2019.

[110] Song Mei, Yu Bai, and Andrea Montanari. The landscape of empirical risk for non-convex losses. *arXiv preprint arXiv:1607.06534*, 2016.

[111] Reshef Meir and David Parkes. On sex, evolution, and the multiplicative weights update algorithm. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 929–937. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[112] Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. Cycles in adversarial regularized learning. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2703–2717, 2018.

[113] Panayotis Mertikopoulos, Houssam Zenati, Bruno Lecouat, Chuan-Sheng Foo, Vijay Chandrasekhar, and Georgios Piliouras. Mirror descent in saddle-point problems: Going the extra (gradient) mile. *CoRR*, abs/1807.02629, 2018.

[114] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? *arXiv preprint arXiv:1801.04406*, 2018.

[115] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of GANs. In *Advances in Neural Information Processing Systems*, pages 1825–1835, 2017.

[116] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

[117] Gerard GL Meyer. Accelerated Frank-Wolfe algorithms. *SIAM Journal on Control*, 1974.

[118] Aryan Mokhtari, Asuman Ozdaglar, and Sarath Pattathil. A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. *arXiv preprint arXiv:1901.08511*, 2019.

[119] Jorge J Moré and Danny C Sorensen. On the use of directions of negative curvature in a modified newton method. *Mathematical Programming*, 16(1):1–20, 1979.

[120] Mohammad Sal Moslehian. Ky fan inequalities. *CoRR*, abs/1108.1467, 2011.

[121] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent GAN optimization is locally stable. In *Advances in Neural Information Processing Systems*, pages 5585–5595, 2017.

[122] Ricardo Ñanculef, Emanuele Frandi, Claudio Sartori, and Héctor Allende. A novel Frank-Wolfe algorithm. analysis and applications to large-scale SVM training. *Information Sciences*, 2014.

[123] A. Nedic and A. Ozdaglar. Subgradient methods for saddle-point problems. *J Optim Theory Appl*, 142:205–228, 2009.

[124] Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 2012.

[125] Yurii Nesterov. *Introductory lectures on convex optimization*, volume 87. Springer Science and Business Media, 2004.

[126] Yurii Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.

[127] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013.

[128] Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[129] Julie Nutini, Mark Schmidt, Issam Laradji, Michael Friedlander, and Hoyt Koepke. Coordinate descent converges faster with the Gauss-Southwell rule than random selection. In *International Conference on Machine Learning*, 2015.

[130] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[131] Gerasimos Palaiopanos, Ioannis Panageas, and Georgios Piliouras. Multiplicative weights update with constant step-size in congestion games: Convergence, limit cycles and chaos. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5874–5884, 2017.

[132] Ioannis Panageas and Georgios Piliouras. Average case performance of replicator dynamics in potential games via computing regions of attraction. *17th ACM Conference on Economics and Computation (EC)*, http://arxiv.org/abs/1403.3885, 2016.

[133] Ioannis Panageas and Georgios Piliouras. Gradient descent only converges to minimizers: Non-isolated critical points and invariant regions. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 2:1–2:12, 2017.

[134] Robin Pemantle. Nonconvergence to unstable points in urn models and stochastic approximations. *The Annals of Probability*, pages 698–712, 1990.

[135] Lawrence Perko. *Differential Equations and Dynamical Systems*. Springer, 3nd. edition, 1991.

[136] Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 2010.

[137] Hassan Rafique, Mingrui Liu, Qihang Lin, and Tianbao Yang. Non-convex min-max optimization: Provable algorithms and applications in machine learning. *arXiv preprint arXiv:1810.02060*, 2018.

[138] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, 2008.

[139] Alexander Rakhlin and Karthik Sridharan. Online learning with predictable sequences. In *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, pages 993–1019, 2013.

[140] A Ravindran, Gintaras Victor Reklaitis, and Kenneth Martin Ragsdell. *Engineering optimization: methods and applications*. John Wiley & Sons, 2006.

[141] Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. Stochastic frank-wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2016.

[142] J. Robinson. An iterative method of solving a game. In *Annals of Mathematics*, pages 296–301, 1951.

[143] William H Sandholm. Evolutionary game theory. In *Encyclopedia of Complexity and Systems Science*, pages 3176–3205. Springer, 2009.

[144] Maziar Sanjabi, Jimmy Ba, Meisam Razaviyayn, and Jason D Lee. On the convergence and robustness of training GANs with regularized optimal transport. In *Advances in Neural Information Processing Systems*, pages 7091–7101, 2018.

[145] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 2013.

[146] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[147] Anshumali Shrivastava and Ping Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *Advances in Neural Information Processing Systems*, 2014.

[148] Michael Shub. *Global Stability of Dynamical Systems*. Springer Science & Business Media, 1987.

[149] Michael Shub. *Global Stability of Dynamical Systems*. Springer-Verlag, 1987.

[150] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

[151] Sören Sonnenburg and Vojtech Franc. Coffin: A computational framework for linear svms. In *ICML*, 2010.

[152] Michael Spivak. *Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus*. Addison-Wesley, 1965.

[153] Ju Sun, Qing Qu, and John Wright. Complete dictionary recovery over the sphere ii: Recovery by riemannian trust-region method. *arXiv preprint arXiv:1511.04777*, 2015.

[154] Vasilis Syrgkanis, Alekh Agarwal, Haipeng Luo, and Robert E. Schapire. Fast convergence of regularized learning in games. In *Annual Conference on Neural Information Processing Systems 2015*, pages 2989–2997, 2015.

[155] Rashish Tandon, Qi Lei, Alexandros G Dimakis, and Nikos Karampatziakis. Gradient coding: Avoiding stragglers in distributed learning. In *International Conference on Machine Learning*, pages 3368–3376, 2017.

[156] Joel A Tropp et al. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015.

[157] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[158] Arnaud Vandaele, Nicolas Gillis, Qi Lei, Kai Zhong, and Inderjit Dhillon. Efficient and non-convex coordinate descent for symmetric nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 64(21):5571–5584, 2016.

[159] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[160] J Von Neumann. Zur theorie der gesellschaftsspiele. In *Math. Ann.*, pages 295–320, 1928.

[161] Jialei Wang and Lin Xiao. Exploiting strong convexity from data with primal-dual first-order algorithms. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3694–3702. JMLR. org, 2017.

[162] Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*, 2017.

[163] Andrés Weintraub, Carmen Ortiz, and Jaime González. Accelerating convergence of the Frank-Wolfe algorithm. *Transportation Research Part B: Methodological*, 1985.

[164] Jay Whang, Qi Lei, and Alexandros G Dimakis. Compressed sensing with invertible generative models and dependent noise. *arXiv preprint arXiv:2003.08089*, 2020.

[165] Lingfei Wu, Ian En-Hsu Yen, Jinfeng Yi, Fangli Xu, Qi Lei, and Michael Witbrock. Random warping series: A random features method for time-series embedding. *arXiv preprint arXiv:1809.05259*, 2018.

[166] Shanshan Wu, Alexandros G Dimakis, and Sujay Sanghavi. Learning distributions generated by one-layer ReLU networks. *arXiv preprint arXiv:1909.01812*, 2019.

[167] Lin Xiao, Adams Wei Yu, Qihang Lin, and Weizhu Chen. Dscovr: Randomized primal-dual block coordinate algorithms for asynchronous distributed optimization. *Journal of Machine Learning Research*, 2019.

[168] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *Advances in neural information processing systems*, pages 1537–1544, 2005.

[169] Zhewei Yao, Amir Gholami, Qi Lei, Kurt Keutzer, and Michael W Mahoney. Hessian-based analysis of large batch training and robustness to adversaries. In *Advances in Neural Information Processing Systems*, pages 4949–4959, 2018.

[170] Ian En-Hsu Yen, Ting-Wei Lin, Shou-De Lin, Pradeep K Ravikumar, and Inderjit S Dhillon. Sparse random feature algorithm as coordinate descent in hilbert space. In *Advances in Neural Information Processing Systems*, pages 2456–2464, 2014.

[171] Jinfeng Yi, Qi Lei, Wesley M Gifford, Ji Liu, Junchi Yan, and Bowen Zhou. Fast unsupervised location category inference from highly inaccurate mobility data. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 55–63. SIAM, 2019.

[172] Jinfeng Yi, Qi Lei, Junchi Yan, and Wei Sun. Session expert: A lightweight conference session recommender system. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1677–1682. IEEE, 2018.

[173] Adams Wei Yu, Qihang Lin, and Tianbao Yang. Doubly stochastic primal-dual coordinate method for empirical risk minimization and bilinear saddle-point problem. *arXiv preprint arXiv:1508.03390*, 2015.

[174] Hsiang-Fu Yu, Cho-Jui Hsieh, Qi Lei, and Inderjit S Dhillon. A greedy approach for budgeted maximum inner product search. In *Advances in Neural Information Processing Systems*, pages 5453–5462, 2017.

[175] Jiong Zhang, Qi Lei, and Inderjit S Dhillon. Stabilizing gradients for deep neural networks via efficient svd parameterization. *arXiv preprint arXiv:1803.09327*, 2018.

[176] Pengchuan Zhang, Qiang Liu, Dengyong Zhou, Tao Xu, and Xiaodong He. On the discrimination-generalization tradeoff in GANs. 2018.

[177] Yuchen Zhang, Xi Chen, Denny Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In *NIPS*, pages 1260–1268, 2014.

[178] Yuchen Zhang and Lin Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *arXiv preprint arXiv:1409.3257*, 2014.

[179] Jiacheng Zhuo, Qi Lei, Alexandros G Dimakis, and Constantine Caramanis. Communication-efficient asynchronous stochastic frank-wolfe over nuclear-norm balls. *arXiv preprint arXiv:1910.07703*, 2019.

[180] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 2005.

Vita

Qi Lei was born in Nanjing, China on 5th of June 1992. She received the Bachelor of Science degree in Mathematics from Zhejiang University in 2014. She was admitted to Oden Institute from the University of Texas at Austin in August, 2014 and started graduate studies since then. She was a research fellow at Simons Institute for the Foundations of Deep Learning Program in Summer 2019. Afterwards she also visited Princeton University/IAS for one year from September 2019 to June 2020. Her main research interests are in machine learning, deep learning and optimization. Qi has received several awards, including four years of the National Initiative for Modeling and Simulation Graduate Research Fellowship, and Simons-Berkeley Research Fellowship for 2019 summer. She also owns several patents.

Permanent address: 3365 Lake Austin Blvd
Austin, Texas 78703

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.