EPFL

Thèse n° 9613

Augmented Lagrangian Methods for Provable and Scalable Machine Learning

Présentée le 27 avril 2023

Faculté des sciences et techniques de l'ingénieur
Laboratoire de systèmes d'information et d'inférence
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Mehmet Fatih SAHIN

Acceptée sur proposition du jury

Prof. M. Jaggi, président du jury
Prof. V. Cevher, directeur de thèse
Prof. N. He, rapporteuse
Dr S. Lu, rapporteur
Prof. N. Boumal, rapporteur

École polytechnique fédérale de Lausanne

2023

To my dear family…

Acknowledgements

I would like to express my gratitude to those who have supported me during this challenging journey. I am especially thankful to my supervisor, Volkan Cevher, for giving me the chance to complete my Ph.D. in such a renowned country, school, and research lab surrounded by brilliant and inspiring individuals from diverse fields. I am particularly grateful for his patience and understanding towards me. This experience has been a valuable lesson, not just in research and optimization, but also in life as a whole.

I am deeply grateful to Songtao Lu, Nicolas Boumal, Niao He, and Martin Jaggi for serving on my thesis committee. Their insightful comments and suggestions greatly helped me improve the quality of my writing in this thesis.

I am thankful to Alp and Baran for their constant support during the early stages of my doctoral journey. Their guidance and encouragement provided a steady source of strength during challenging times. Ahmet has always been a very good friend who inspired me throughout my Ph.D. with his constant dedication to his work. I regret that I was unable to be a frequent gym companion for him.

Attending conferences is undoubtedly one of the most rewarding experiences of pursuing a PhD, and having a friend like Bang who can add a touch of humor to the whole journey can make the experience all the more enjoyable. I am also grateful to him for promptly responding to my questions about mathematical concepts even after departing from our group.

I want to extend my heartfelt thanks to Gosia, our lab admin, for being an immense help to me on many occasions, whether it was work-related or for making my life in Lausanne easier. I will always miss the lengthy discussions that we had in the office, taking a break from our work and delving into topics ranging from religion to politics.

I had amazing colleagues during my time at LIONS. I would like to thank Grigorios, Fanghui, Elias, Ahmet and Ya-Ping for the collaborating with me and for their contributions to my work. Additionally, I would like to extend my thanks to Yura Malitsky, Ali Kavis, Thomas S., Thomas P., Leelo, Fabian, Igor, Arda, Maria, Nadav, Pedro, Ali Ramezani, Kimon, Paul, Stratis, Luca, and all those who have touched various aspects of my journey. I would like to particularly thank Thomas S. who helped me with the french version of the abstract of this thesis.

I would like to extend my sincerest appreciation to my parents Safiye and Murat, my sisters Sevdenur and Sumeyye for their unconditional love, encouragement, and support throughout my life. Knowing that they are always behind me has given me the strength and motivation to persevere.

Acknowledgements

Last but not the least, I would like to express my deepest appreciation to the love of my life, Saniye, who has been with me all the time through my Ph.D. journey. The start of our relation almost dates back to the beginning of my Ph.D. and she has been a constant source of support and comfort throughout this challenging and often difficult journey. I am grateful to her for standing by my side and for being there for me, even during my grumpiest and most stressful moments. I would like to dedicate this thesis to my dear family, including my beloved son, Merih, who has recently become a part of our lives.

Lausanne, April 21, 2023 M. F. S.

Abstract

Non-convex constrained optimization problems have become a powerful framework for modeling a wide range of machine learning problems, with applications in k-means clustering, large-scale semidefinite programs (SDPs), and various other tasks. As the performance of these methods on real-world tasks has shown promising results, the need to develop efficient algorithms and understand their theoretical properties has become increasingly important. Augmented Lagrangian methods provide strong tools for solving constrained problems by breaking them down into a sequence of relatively easier unconstrained subproblems. Theoretical properties of these methods have been extensively studied for problems where both the objective function and constraints are convex. Additionally, there have been efforts to provide convergence guarantees to the first-order stationary points of constrained problems when only the objective function is non-convex. However, the scenario in which the objective function and constraints are both non-convex has yet to be sufficiently explored. This thesis is dedicated to the development and theoretical analysis of algorithms that are grounded in the Augmented Lagrangian method, with an emphasis on efficiency and practicality.

First, we introduce a generic inexact Augmented Lagrangian framework that enables the use of either first-order or second-order subsolvers in the subproblems to attain convergence guarantees for the first (or second) order stationary points of the original constrained problem. This is accomplished with an algorithm that can be implemented practically. The success of the algorithm relies on a verifiable geometric regularity condition. We showcase the effectiveness of the algorithm via a range of numerical examples, including k-means clustering and ℓ_∞ image denoising problem that incorporates a generative adversarial network (GAN) prior to counteract adversarial examples.

Next, we direct our attention to a more specialized algorithm aimed at solving semidefinite programming (SDP) problems by factorizing the decision variable. The resulting optimization problems are inherently non-convex and nonlinear. We obtain the rate of convergence with an augmented Lagrangian method which combines aspects of both linearized and inexact augmented Lagrangian methods.

Finally, we present a linearized alternating direction method of multipliers (ADMM) framework that is more practical and requires only two consecutive proximal gradient steps on the primal variables and a gradient ascent step in the dual. This framework is designed to address the increasingly important problem of minimizing two sets of variables with a separable non-convex composite objective that has nonlinear constraints. Our analysis enables us to recover known convergence rates with a single loop algorithm, which is less complex than the inexact Augmented

Abstract

Lagrangian variants. Furthermore, our framework accommodates the linearized augmented Lagrangian as a special case. The numerical evidence on various machine learning tasks, including causal learning, clustering and maximum cut problems, illustrates that the proposed algorithm is versatile, scalable and accurate while requiring minimal tuning.

Key words: non-convex optimization, nonlinearly constrained optimization, SDPs, augmented Lagrangian methods, first-order methods, practical algorithms, KKT points.

Résumé

Les problèmes d'optimisation non convexe sous contrainte sont devenus un outil puissant pour modéliser un large éventail de problèmes d'apprentissage automatique, avec des applications dans le partitionnement k-moyennes, les programmes semi-définis (SDP) à grande échelle et diverses autres tâches. Puisque les performances de ces méthodes sur des tâches réelles ont montré des résultats prometteurs, la nécessité de développer des algorithmes efficaces et de comprendre leurs propriétés théoriques est devenue de plus en plus importante. Les méthodes de Lagrangien augmenté fournissent des moyens puissants pour résoudre des problèmes contraints en les décomposant en une séquence de sous-problèmes non-contraints plus faciles à résoudre. Les propriétés théoriques de ces méthodes ont été largement étudiées pour les problèmes où la fonction objectif et les contraintes sont convexes. En outre, des efforts ont été déployés pour fournir des garanties de convergence aux points stationnaires du premier ordre des problèmes contraints lorsque seule la fonction objectif est non convexe. Cependant, le scénario dans lequel la fonction objective et les contraintes sont toutes deux non-convexes n'a pas encore été suffisamment exploré. Cette thèse est consacrée au développement et à l'analyse théorique d'algorithmes basés sur la méthode du lagrangien augmenté, en mettant l'accent sur l'efficacité et la praticité.

Tout d'abord, nous présentons une méthode générique de Lagrangien augmenté inexact qui permet l'utilisation de solveurs du premier ou du second ordre dans les sous-problèmes pour obtenir des garanties de convergence aux points stationnaires du premier (ou du second) ordre du problème contraint d'origine. Cela est réalisé avec un algorithme qui peut être implémenté en pratique. Le succès de l'algorithme repose sur une condition de régularité géométrique vérifiable. Nous démontrons l'efficacité de l'algorithme à l'aide d'une série d'exemples numériques, notamment le partitionnement k-moyennes et le problème de débruitage d'image ℓ_∞ qui utilise un réseau antagoniste génératif (GAN) comme a priori afin de contrecarrer les exemples contradictoires (*adversarial examples*).

Ensuite, nous nous intéressons à un algorithme plus spécialisé visant à résoudre les problèmes de programmation semi-définie (SDP) en factorisant la variable de décision. Les problèmes d'optimisation qui en résultent sont intrinsèquement non convexes et non linéaires. Nous obtenons un taux de convergence avec une méthode de Lagrangien augmenté qui combine des aspects des méthodes de Lagrangien augmenté linéarisées et inexactes.

Enfin, nous présentons une linéarisation de la méthode des multiplicateurs de direction alternée (ADMM) qui est plus pratique et ne nécessite que deux étapes consécutives de gradient proximal sur les variables primales et une étape d'ascension de gradient dans le dual. Cette méthode est conçue pour traiter le problème de plus en plus important de la minimisation de deux ensembles

de variables avec un objectif composite non convexe séparable qui a des contraintes non linéaires. Notre analyse nous permet de retrouver les taux de convergence connus avec un algorithme à boucle unique, qui est moins complexe que les variantes inexactes du lagrangien augmenté. En outre, notre méthode présente comme cas particulier le Lagrangien augmenté linéarisé. Les résultats numériques obtenus sur diverses tâches d'apprentissage automatique, y compris l'apprentissage causal, le partitionnement et les problèmes de coupe maximum, montrent que l'algorithme proposé est polyvalent, évolutif et précis, tout en nécessitant un réglage minimal.

Mots clés : optimisation non convexe, optimisation contrainte non linéaire, programmes semi-définis, méthodes de Lagrangien augmenté, méthodes de premier ordre, algorithmes pratiques, points KKT.

Bibliographic Note

This dissertation is based on the following publications:

- Mehmet Fatih Sahin, Armin Eftekhari, Ahmet Alacaoglu, Fabian Latorre and Volkan Cevher. "An Inexact Augmented Lagrangian Framework for Non-convex Optimization with Nonlinear Constraints", NeurIPS, 2019
- Cong Vu Bang, Ahmet Alacaoglu, Mehmet Fatih Sahin, Alp Yurtsever and Volkan Cevher. "A First-order Augmented Lagrangian Framework for Non-convex Optimization with Nonlinear Constraints", Workshop on Modern Trends in Non-convex Optimization for Machine Learning, ICML, 2018
- Mehmet Fatih Sahin, Armin Eftekhari, Olivier Fercoq and Volkan Cevher. "A Linearized Alternating Direction Method of Multipliers Framework for Non-convex Problems with Nonlinear Constraints", Work under review.

At the end of some chapters we include a “Bibliographic Notes” section to distinguish the contributions of the author of this dissertation in the abovementioned publications. My other publications which are not included in the dissertation:

- Ehsan Asadi Kangarshahi, Ya-Ping Hsieh, Mehmet Fatih Sahin and Volkan Cevher. "Let’s be honest: An Optimal No-regret Framework for Zero-sum Games", ICML, 2018
- Elias Abad Rocamora, Mehmet Fatih Sahin, Fanghui Liu, Grigorios Chrysos and Volkan Cevher. "Sound and Complete Verification of Polynomial Networks", NeurIPS, 2022
- Elias Abad Rocamora, Mehmet Fatih Sahin, Fanghui Liu, Grigorios Chrysos and Volkan Cevher. "α-convexification for Sound and Complete Verification of Twice-differentiable Classifiers", Work in progress.

Contents

1 Introduction

Non-convex optimization has gained popularity over the past two decades due to its ability to effectively model complex real world scenarios in the field of deep learning and machine learning. As a result, there has been a growing need to develop effective and practical algorithms capable of solving these types of problems. A considerable body of research has concentrated on non-convex optimization problems without any constraints or featuring constraints that are relatively simple to handle [TH12, Zei12, KB14, Doz16, RKK19, CO19]. However, in some cases, it becomes essential to incorporate non-convex constraints along with non-convex objectives to accurately capture the complexities of learning tasks and improve the overall model. For instance, due to their data-driven nature, machine learning models might produce biased results towards some underrepresented groups and therefore cause discrimination. Optimizing these models with non-convex *fairness* constraints [CJG+19] is one way to mitigate this problem.

Semidefinite programming (SDP) is a type of optimization problem that involves minimizing a linear objective function subject to linear constraints and a semidefinite constraint. In discrete optimization, some of the most powerful relaxations for important problems such as maximum-cut [GW95] and community detection [ABKK17] rely on semidefinite programming. SDPs also provide a powerful framework for achieving certifiable robustness in deep neural networks [FMP20]. While algorithms designed to solve SDPs to arbitrary accuracy within polynomial time offer promising theoretical results, they often face significant challenges when it comes to practical implementation, particularly in terms of scalability. As the size of the problem grows larger, both the computational time and memory requirements for solving SDPs can become prohibitively high, making it difficult to work with these algorithms in practice. This highlights the ongoing need for developing more efficient and scalable approaches to solving SDPs [YTF+21, YUTC17], particularly in the context of larger and more complex optimization problems. The Burer-Monteiro (BM) factorization is a technique that can be used to efficiently solve certain types of SDP problems, by approximating the semidefinite constraint with a low-rank matrix. This technique has been shown to be effective in solving large-scale SDP problems in machine learning [BM03b, BVB16] and other fields. However, the optimization problem arising from BM factorization is inherently challenging, as it involves both a non-convex objective

function and non-convex constraints. Therefore, it is essential to devise approaches with provable guarantees to effectively address these complex optimization problems.

While there are numerous algorithms available for addressing these more complex class of problems, it is crucial to consider factors such as efficiency, scalability, and ease of implementation to ensure that they can be effectively utilized in real-world situations. A substantial portion of these algorithms, despite providing theoretical convergence guarantees, suffer from impracticality or depend on assumptions that prove challenging to verify [BST18, CGT18, BGM+16]. Finding the right balance between these factors remains an active area of research in the field of nonlinear optimization. Therefore, we pose the following research question:

Is it possible to design algorithms which are both practical and provable with verifiable assumptions?

In this thesis, we answer this affirmatively and develop various algorithms for non-convex constrained optimization which are based on Augmented Lagrangian method.

1.1 Problem formulation

We consider the following optimization problem,

$$\min_{x\in\mathbb{R}^d} f(x)+g(x) \qquad \text{s.t.} \qquad A(x)=0, \tag{1.1}$$

where $f:\mathbb{R}^d\to\mathbb{R}$ is a twice differentiable function whose gradient is given by $\nabla f(x)\in\mathbb{R}^d$; $g:\mathbb{R}^d\to\mathbb{R}\cup\{\infty\}$ is a proximal-friendly, (possibly) non-differentiable, proper, closed and convex function; $A:\mathbb{R}^d\to\mathbb{R}^m$ is a twice-differentiable mapping whose Jacobian is denoted as $\mathbf{D}A(x)\in\mathbb{R}^{m\times d}$. We additionally assume f and A satisfies the smoothness property such that

$$\|\nabla f(x)-\nabla f(x')\|\le\lambda_f\|x-x'\|, \quad \|\mathbf{D}A(x)-\mathbf{D}A(x')\|\le\lambda_A\|x-x'\|, \tag{1.2}$$

for $\lambda_f,\lambda_A\ge 0$ and every $x,x'\in\mathbb{R}^d$.

Solving non-convex optimization problems with constraints presents a persistent challenge in the field of machine learning. When the constraints are either convex or linear, several works have explored and developed various algorithms for solving this class of problems. Some of these algorithms include primal-dual algorithms [HLR16, HH19, ZYZ22], inexact augmented Lagrangian methods [Xu21], and trust-region approaches [CGT11], among others.

1.1.1 Augmented Lagrangian method

Augmented Lagrangian method is a classical algorithm, which first appeared in [Hes69, Pow69] and extensively studied afterwards in [Ber82, BM14]. The use of Augmented Lagrangian methods holds a prominent place in the field of optimization, as they effectively address constrained optimization problems by melding the strengths of the Lagrangian method and penalty methods. The combination of these techniques results in more accurate and efficient solutions to complex problems in a variety of disciplines, including engineering [LFJ11], economics, and science. The Augmented Lagrangian approach integrates the penalty function into the Lagrangian, leading to increased stability and convergence in iterative algorithms. This robustness enables the handling of non-convex and non-differentiable constraints, making it a versatile and effective tool in addressing real-world challenges. Additionally, these methods have also been successful in decomposing large-scale problems into smaller subproblems, facilitating parallel computing and solving problems in distributed systems [CDZ15]. For solving (1.1), ALM suggests solving the problem

$$\min_x \max_y \mathscr{L}_\beta(x, y) + g(x), \tag{1.3}$$

where, for penalty weight $\beta > 0$, $\mathscr{L}_\beta$ is the corresponding augmented Lagrangian, defined as

$$\mathscr{L}_\beta(x, y) := f(x) + \langle A(x), y\rangle + \frac{\beta}{2}\|A(x)\|^2. \tag{1.4}$$

The minimax formulation in (1.3) naturally suggests the following algorithm for solving (1.1):

$$x_{k+1} \in \operatorname*{argmin}_x \mathscr{L}_\beta(x, y_k) + g(x), \tag{1.5}$$

$$y_{k+1} = y_k + \sigma_k A(x_{k+1}),$$

where the dual step sizes are denoted as $\{\sigma_k\}_k$. However, computing x_{k+1} above requires solving the non-convex problem (1.5) to optimality, which is typically intractable. Instead, it is often easier to find an approximate first- or second-order stationary point of (1.5).

Hence, we argue that by gradually improving the stationarity precision and increasing the penalty weight β above, we can reach to a stationary point of the main constrained problem, as detailed in Section 2.3.

1.1.2 Notations

We use the notations $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ for the standard inner product and norm on $\mathbb{R}^d$, respectively. For matrices, $\|\cdot\|$ and $\|\cdot\|_F$ denote the spectral and Frobenius norms, respectively. For the convex function $g : \mathbb{R}^d \to \mathbb{R}$, the subdifferential set at $x \in \mathbb{R}^d$ is denoted by $\partial g(x)$ and we will occasionally use the notation $\partial g(x)/\beta = \{z/\beta : z \in \partial g(x)\}$. When presenting iteration complexity results, we often use $\widetilde{O}(\cdot)$ which suppresses the logarithmic dependencies. Gradient of differentiable $f : \mathbb{R}^d \to \mathbb{R}$ at x is denoted by $\nabla f(x)$. For an operator $A : \mathbb{R}^d \to \mathbb{R}^m$ with components $\{A_i\}_{i=1}^m$,

$DA(x) \in \mathbb{R}^{m\times d}$ denotes the Jacobian of A, where the ith row of $DA(x)$ is the vector $\nabla A_i(x) \in \mathbb{R}^d$. Given $x \in \mathbb{R}^d$ and $\gamma > 0$, the proximal operator, $P_{\gamma,g} : \mathbb{R}^d \to \mathbb{R}^d$, associated to g takes the form $P_{\gamma,g}(x) = \operatorname{argmin}_y g(y) + \frac{1}{2\gamma}\|x - y\|^2$. We denote $\delta_{\mathscr{X}} : \mathbb{R}^d \to \mathbb{R}$ as the indicator function of a set $\mathscr{X} \subset \mathbb{R}^d$. An integer interval is denoted by $[k_0 : k_1] = \{k_0, \cdots, k_1\}$ for integers $k_0 \le k_1$.

1.1.3 Optimality conditions

First-order necessary optimality conditions for (1.1) are well-studied. Indeed, $x \in \mathbb{R}^d$ is a first-order stationary point of (1.1) if there exists $y \in \mathbb{R}^m$ such that

$$-\nabla_x \mathscr{L}_\beta(x, y) \in \partial g(x), \qquad A(x) = 0, \tag{1.6}$$

which is in turn the necessary optimality condition for (1.3). Inspired by this, we say that x is an (ϵ_f, β) first-order stationary point of (1.3) if there exists a $y \in \mathbb{R}^m$ such that

$$\operatorname{dist}(-\nabla_x \mathscr{L}_\beta(x, y), \partial g(x)) \le \epsilon_f, \qquad \|A(x)\| \le \epsilon_f, \tag{1.7}$$

for $\epsilon_f \ge 0$. In light of (1.7), a metric for evaluating the stationarity of a pair $(x, y) \in \mathbb{R}^d \times \mathbb{R}^m$ is

$$\operatorname{dist}\big(-\nabla_x \mathscr{L}_\beta(x, y), \partial g(x)\big) + \|A(x)\|, \tag{1.8}$$

which we use as the first-order stopping criterion. As an example, for a convex set $\mathscr{X} \subset \mathbb{R}^d$, suppose that $g = \delta_{\mathscr{X}}$ is the indicator function on $\mathscr{X}$. Let also $T_{\mathscr{X}}(x) \subseteq \mathbb{R}^d$ denote the tangent cone to $\mathscr{X}$ at x, and with $P_{T_{\mathscr{X}}(x)} : \mathbb{R}^d \to \mathbb{R}^d$ we denote the orthogonal projection onto this tangent cone. Then, for $u \in \mathbb{R}^d$, it is not difficult to verify that

$$\operatorname{dist}\big(u, \partial g(x)\big) = \|P_{T_{\mathscr{X}}(x)}(u)\|. \tag{1.9}$$

When $g = 0$, a first-order stationary point $x \in \mathbb{R}^d$ of (2.1) is also second-order stationary if

$$\lambda_{\min}(\nabla_{xx} \mathscr{L}_\beta(x, y)) \ge 0, \tag{1.10}$$

where $\nabla_{xx}\mathscr{L}_\beta$ is the Hessian of $\mathscr{L}_\beta$ with respect to x, and $\lambda_{\min}(\cdot)$ returns the smallest eigenvalue of its argument. Analogously, x is an $(\epsilon_f, \epsilon_s, \beta)$ second-order stationary point if, in addition to (1.7), it holds that

$$\lambda_{\min}(\nabla_{xx} \mathscr{L}_\beta(x, y)) \ge -\epsilon_s, \tag{1.11}$$

for $\epsilon_s \ge 0$. Naturally, for second-order stationarity, we use $\lambda_{\min}(\nabla_{xx}\mathscr{L}_\beta(x, y))$ as the stopping criterion.

Gradient mapping (as in [GLZ16]), defined below, plays an important role in our convergence analysis. However, by itself, it does not correspond to any standard measure of stationarity for the problem (1.1) as explained in the sequel.

Definition 1. ***(Gradient mapping)*** *Given $y \in \mathbb{R}^d$ and $\gamma > 0$, the gradient mapping $G_{\beta,\gamma}(\cdot; y)$:*

$\mathbb{R}^d \to \mathbb{R}^d$ *takes* $x \in \mathbb{R}^d$ *to*

$$G_{\beta,\gamma}(x,y) = \frac{x - x^+}{\gamma}, \tag{1.12}$$

where $x^+ = P_{\gamma,g}(x - \gamma\nabla_x \mathscr{L}_\beta(x,y))$. *If we set* $g \equiv 0$ *in* (1.1)*, the gradient mapping reduces to* $G_{\beta,\gamma}(x,y) = \nabla_x \mathscr{L}_\beta(x,y)$.

Note also from optimality conditions of the proximal mapping that

$$G_{\beta,\gamma}(x,y) \in \nabla f(x) + \mathbf{D}A(x)^\top \tilde{y} + \partial g(x^+), \tag{1.13}$$

where $\tilde{y} = y + \beta A(x)$. Here, $f(\cdot) + \mathbf{D}A(\cdot)^\top \tilde{y}$ and $\partial g(\cdot)$ are calculated at different points. To resolve this issue, we propose to use the displaced gradient mapping as a meaningful metric for showing approximate stationarity

$$\widehat{G_{\beta,\gamma}}(x^+, \tilde{y}) := G_{\beta,\gamma}(x,y) + \nabla f(x^+) - \nabla f(x) + \big(\mathbf{D}A(x^+) - \mathbf{D}A(x)\big)^\top \tilde{y}.$$

Therefore, a linear combination of $\|\widehat{G_{\beta,\gamma}}(x^+,\tilde{y})\|^2$ and the feasibility gap $\|A(x^+)\|^2$ is a natural metric to measure the first-order stationarity of a pair $(x^+,\tilde{y})$ in problem (1.1).

1.2 Contributions and organization

In this dissertation, we focus on revealing *verifiable assumptions on the constraint set* for problems of the form (1.1) to be able to *provide convergence guarantees* to the *practical and implementable algorithms* we propose. Our goal is to bridge the gap between theory and practice. To achieve this goal:

In **Chapter 2**,

- We propose a *practical* inexact augmented Lagrangian method (iALM) for nonconvex problems with nonlinear constraints.
- We characterize the total computational complexity of our method subject to a verifiable geometric condition, which is closely related to the Polyak-Lojasiewicz and Mangasarian-Fromowitz conditions.
- In particular, when a first-order solver is used for the inner iterates, we prove that iALM finds a first-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^4)$ calls to the first-order oracle.
- If, in addition, the problem is smooth and a second-order solver is used for the inner iterates, iALM finds a second-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^5)$ calls to the second-order oracle, which matches the known theoretical complexity result in the literature.
- We also provide strong numerical evidence on large-scale machine learning problems, including the Burer-Monteiro factorization of semidefinite programs, and a novel nonconvex relaxation of the standard basis pursuit template.

- For these examples, we also show how to *verify* our geometric condition.

In **Chapter 3**,

- We consider a canonical nonlinear-constrained nonconvex problem arising from factorizing the decision variable in semidefinite programming examples (SDPs).
- We propose a simple primal-dual splitting scheme that provably converges to a stationary point of the non-convex problem.
- We achieve this desideratum via an adaptive and inexact augmented Lagrangian method.
- The new algorithm features a slow $\mathcal{O}(1/\epsilon^6)$ convergence rate, which it counteracted by its cheap per-iteration complexity. We provide numerical evidence on large-scale machine learning problems, modeled typically via semidefinite relaxations.

In **Chapter 4**,

- We study an increasingly important problem of minimizing two sets of variables with the separable nonconvex composite objective with nonlinear constraints.
- We introduce a linearized alternating direction method of multipliers (ADMM) framework that only requires two consecutive proximal gradient steps on the primal variables and a gradient ascent step in the dual.
- The proposed scheme achieves $\epsilon-$first order stationarity by reducing the feasibility and gradient mapping at a rate $\tilde{\mathcal{O}}(\frac{1}{\epsilon^4})$ subject to a regularity condition on the constraints.
- We also establish the same complexity result to ϵ approximate KKT points of the constrained problem. Our analysis allows us to recover the known convergence rates with a single loop algorithm, which is simpler than the inexact Augmented Lagrangian variants.
- Our framework also handles the linearized augmented Lagrangian as a special case. Numerical evidence on large-scale nonconvex machine learning problems (such as continuous relaxation of causal learning problem, SDP relaxation of clustering and Max-Cut problems) show that the algorithm is *scalable and accurate* while requiring *very little tuning*.

In short, this thesis explores the development of efficient non-convex optimization methods based on Augmented Lagrangian function, which are well-suited for machine learning applications. We address the challenges of computational complexity and convergence while demonstrating effectiveness on real-world tasks.

2 iALM for Non-convex Problems with Nonlinear constraints

In this chapter, we propose a practical inexact augmented Lagrangian method (iALM) for non-convex problems with nonlinear constraints. We characterize the total computational complexity of our method subject to a verifiable geometric condition, which is closely related to the Polyak-Lojasiewicz and Mangasarian-Fromowitz conditions.

In particular, when a first-order solver is used for the inner iterates, we prove that iALM finds a first-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^4)$ calls to the first-order oracle. If, in addition, the problem is smooth and a second-order solver is used for the inner iterates, iALM finds a second-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^5)$ calls to the second-order oracle, which matches the known theoretical complexity result in the literature.

We also provide strong numerical evidence on large-scale machine learning problems, including the Burer-Monteiro factorization of semidefinite programs, and a novel non-convex relaxation of the standard basis pursuit template. For these examples, we also show how to verify our geometric condition.

2.1 Introduction

We study the non-convex optimization problem

$$\min_{x\in\mathbb{R}^d} f(x) + g(x) \quad \text{s.t.} \quad A(x) = 0, \tag{2.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a continuously-differentiable non-convex function and $A : \mathbb{R}^d \to \mathbb{R}^m$ is a nonlinear operator. We assume that $g : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ is a proximal-friendly convex function [PB$^+$14].

A host of problems in computer science [KN11, Lov03, ZKRW98], machine learning [MNS15, SSGB07], and signal processing [Sin11, SS11] naturally fall under the template (2.1), including max-cut, clustering, generalized eigenvalue decomposition, as well as the quadratic assignment problem (QAP) [ZKRW98].

To solve (2.1), we propose an intuitive and easy-to-implement augmented Lagrangian algorithm, and provide its total iteration complexity under an interpretable geometric condition. Before we elaborate on the results, let us first motivate (2.1) with an application to semidefinite programming (SDP):

Vignette: Burer-Monteiro splitting.A powerful convex relaxation for max-cut, clustering, and many others is provided by the trace-constrained SDP

$$\min_{X\in\mathbb{S}^{d\times d}} \langle C, X\rangle \quad \text{s.t.} \quad B(X) = b,\ \mathrm{tr}(X) \le \alpha,\ X \succeq 0, \tag{2.2}$$

where $C \in \mathbb{R}^{d\times d}$, X is a positive semidefinite $d \times d$ matrix, and $B : \mathbb{S}^{d\times d} \to \mathbb{R}^m$ is a linear operator. If the unique-games conjecture is true, the SDP in equation (2.2) obtains the best possible approximation for the underlying discrete problem [Rag08].

Since d is often large, many first- and second-order methods for solving such SDP's are immediately ruled out, not only due to their high computational complexity, but also due to their storage requirements, which are $\mathcal{O}(d^2)$.

A contemporary challenge in optimization is therefore to solve SDPs using little space and in a scalable fashion. The recent homotopy conditional gradient method, which is based on linear minimization oracles (LMOs), can solve (2.2) in a small space via sketching [YFLC18]. However, such LMO-based methods are extremely slow in obtaining accurate solutions.

A different approach for solving (2.2), dating back to [BM03b, BM05], is the so-called Burer-Monteiro (BM) factorization $X = UU^\top$, where $U \in \mathbb{R}^{d\times r}$ and r is selected according to the guidelines in [Pat98, Bar95], which is tight [WW18]. The BM factorization leads to the following non-convex problem in the template (2.1):

$$\min_{U\in\mathbb{R}^{d\times r}} \langle C, UU^\top\rangle \quad \text{s.t.} \quad B(UU^\top) = b,\ \|U\|_F^2 \le \alpha, \tag{2.3}$$

The BM factorization does not introduce any extraneous local minima [BM05]. Moreover, [BVB16] establishes the connection between the local minimizers of the factorized problem (2.3) and the global minimizers for (2.2). To solve (2.3), the inexact Augmented Lagrangian method (iALM) is widely used [BM03b, BM05, KSP07], due to its cheap per iteration cost and its empirical success.

Every (outer) iteration of iALM calls a solver to solve an intermediate augmented Lagrangian subproblem to near stationarity. The choices include first-order methods, such as the proximal gradient descent [PB$^+$14], or second-order methods, such as the trust region method and BFGS [NW06].[1]

Unlike its convex counterpart [NNTD14, LM16, Xu17b], the convergence rate and the complexity

[1] BFGS is in fact a quasi-Newton method that emulates second-order information.

of iALM for (2.3) are not well-understood, see Section 2.5 for a review of the related literature. Indeed, addressing this important theoretical gap is one of the contributions of our work. In addition,

- We derive the convergence rate of iALM to first-order optimality for solving (2.1) or second-order optimality for solving (2.1) with $g = 0$, and find the total iteration complexity of iALM using different solvers for the augmented Lagrangian subproblems. We provide an extensive comparison with the existing complexity results in optimization, see Section 2.5.
- Our iALM framework is future-proof in the sense that different subsolvers can be substituted.
- We propose a geometric condition that simplifies the algorithmic analysis for iALM, and clarify its connection to well-known Polyak-Lojasiewicz [KNS16] and Mangasarian-Fromovitz [Ber82] conditions. We also verify this condition for key problems in Sections 2.10 and 2.11.

2.2 Preliminaries

Smoothness. We assume smooth $f : \mathbb{R}^d \to \mathbb{R}$ and $A : \mathbb{R}^d \to \mathbb{R}^m$; i.e., there exist $\lambda_f, \lambda_A \geq 0$ s.t.

$$\|\nabla f(x) - \nabla f(x')\| \leq \lambda_f \|x - x'\|, \quad \|DA(x) - DA(x')\| \leq \lambda_A \|x - x'\|, \quad \forall x, x' \in \mathbb{R}^d. \tag{2.4}$$

Smoothness lemma. This next result controls the smoothness of $\mathscr{L}_\beta(\cdot, y)$ for a fixed y. The proof is standard but nevertheless is included in Section 2.9 for completeness.

Lemma 2.2.1 (smoothness). *For fixed $y \in \mathbb{R}^m$ and $\rho, \rho' \geq 0$, it holds that*

$$\|\nabla_x \mathscr{L}_\beta(x, y) - \nabla_x \mathscr{L}_\beta(x', y)\| \leq \lambda_\beta \|x - x'\|, \tag{2.5}$$

for every $x, x' \in \{x'' : \|x''\| \leq \rho, \|A(x'')\| \leq \rho'\}$, where

$$\lambda_\beta \leq \lambda_f + \sqrt{m}\lambda_A \|y\| + (\sqrt{m}\lambda_A \rho' + d\lambda_A'^2)\beta =: \lambda_f + \sqrt{m}\lambda_A \|y\| + \lambda''(A, \rho, \rho')\beta. \tag{2.6}$$

Above, λ_f, λ_A were defined in (2.4) and

$$\lambda_A' := \max_{\|x\| \leq \rho} \|DA(x)\|. \tag{2.7}$$

2.3 Algorithm

To solve the equivalent formulation of (2.1) presented in (1.3), we propose the inexact ALM (iALM), detailed in Algorithm 1. At the k^{th} iteration, Step 2 of Algorithm 1 calls a solver that finds an approximate stationary point of the augmented Lagrangian $\mathscr{L}_{\beta_k}(\cdot, y_k)$ with the accuracy of ϵ_{k+1}, and this accuracy gradually increases in a controlled fashion. The increasing sequence of penalty weights $\{\beta_k\}_k$ and the dual update (Steps 4 and 5) are responsible for continuously enforcing the constraints in (2.1). The appropriate choice for $\{\beta_k\}_k$ will be specified in Corrollary

Sections 2.7.1 and 2.7.2.

The particular choice of the dual step sizes $\{\sigma_k\}_k$ in Algorithm 1 ensures that the dual variable y_k remains bounded.

Algorithm 1 Inexact ALM

Input: Non-decreasing, positive, unbounded sequence $\{\beta_k\}_{k\geq 1}$, stopping thresholds $\tau_f, \tau_s > 0$.
Initialization: Primal variable $x_1 \in \mathbb{R}^d$, dual variable $y_0 \in \mathbb{R}^m$, dual step size $\sigma_1 > 0$.

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

1. **(Update tolerance)** $\epsilon_{k+1} = 1/\beta_k$.
2. **(Inexact primal solution)** Obtain $x_{k+1} \in \mathbb{R}^d$ such that

$$\operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_k}(x_{k+1}, y_k), \partial g(x_{k+1})) \leq \epsilon_{k+1}$$

for first-order stationarity

$$\lambda_{\min}(\nabla_{xx} \mathscr{L}_{\beta_k}(x_{k+1}, y_k)) \geq -\epsilon_{k+1}$$

for second-order-stationarity, if $g = 0$ in (2.1).
3. **(Update dual step size)**

$$\sigma_{k+1} = \sigma_1 \min\Big(\frac{\|A(x_1)\| \log^2 2}{\|A(x_{k+1})\|(k+1)\log^2(k+2)}, 1\Big).$$

4. **(Dual ascent)** $y_{k+1} = y_k + \sigma_{k+1} A(x_{k+1})$.
5. **(Stopping criterion)** If

$$\operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_k}(x_{k+1}), \partial g(x_{k+1})) + \|A(x_{k+1})\| \leq \tau_f,$$

for first-order stationarity and if also $\lambda_{\min}(\nabla_{xx} \mathscr{L}_{\beta_k}(x_{k+1}, y_k)) \geq -\tau_s$ for second-order stationarity, then quit and return x_{k+1} as an (approximate) stationary point of (1.3).

3: **end for**

2.4 Convergence Rate

This section presents the total iteration complexity of Algorithm 1 for finding first and second-order stationary points of problem (1.3). All the proofs are deferred to Section 2.8. Theorem 2.4.1 characterizes the convergence rate of Algorithm 1 for finding stationary points in the number of outer iterations.

Theorem 2.4.1. (convergence rate) *For integers $2 \leq k_0 \leq k_1$, consider the interval $K = [k_0 : k_1]$, and let $\{x_k\}_{k\in K}$ be the output sequence of Algorithm 1 on the interval K.*[2] *Let also $\rho :=$*

[2] The choice of $k_1 = \infty$ is valid here too.

$\sup_{k\in[K]} \|x_k\|$.[3] *Suppose that f and A satisfy (2.4) and let*

$$\lambda'_f = \max_{\|x\|\le\rho} \|\nabla f(x)\|, \qquad \lambda'_A = \max_{\|x\|\le\rho} \|DA(x)\|, \tag{2.8}$$

be the (restricted) Lipschitz constants of f and A, respectively. With $\nu > 0$, assume that

$$\nu\|A(x_k)\| \le \operatorname{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right), \tag{2.9}$$

for every $k \in K$. We consider two cases:

- *If a first-order solver is used in Step 2, then x_k is an $(\epsilon_{k,f}, \beta_k)$ first-order stationary point of (1.3) with*

$$\epsilon_{k,f} = \frac{1}{\beta_{k-1}}\left(\frac{2(\lambda'_f + \lambda'_A y_{\max})(1+\lambda'_A\sigma_k)}{\nu} + 1\right) =: \frac{Q(f,g,A,\sigma_1)}{\beta_{k-1}}, \tag{2.10}$$

  for every $k \in K$, where $y_{\max}(x_1, y_0, \sigma_1) := \|y_0\| + c\|A(x_1)\|$.

- *If a second-order solver is used in Step 2, then x_k is an $(\epsilon_{k,f}, \epsilon_{k,s}, \beta_k)$ second-order stationary point of (1.3) with $\epsilon_{k,s}$ specified above and with*

$$\epsilon_{k,s} = \epsilon_{k-1} + \sigma_k\sqrt{m}\lambda_A\frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu\beta_{k-1}} = \frac{\nu + \sigma_k\sqrt{m}\lambda_A 2\lambda'_f + 2\lambda'_A y_{\max}}{\nu\beta_{k-1}} =: \frac{Q'(f,g,A,\sigma_1)}{\beta_{k-1}}. \tag{2.11}$$

Theorem 2.4.1 states that Algorithm 1 converges to a (first- or second-) order stationary point of (1.3) at the rate of $1/\beta_k$, further specified in Corollary 2.4.2 and Corollary 2.4.3. A few remarks are in order about Theorem 2.4.1.

Regularity. The key geometric condition in Theorem 2.4.1 is (2.9) which, broadly speaking, ensures that the primal updates of Algorithm 1 reduce the feasibility gap as the penalty weight β_k grows. We will verify this condition for several examples in Sections 2.10 and 2.11.

This condition in (2.9) is closely related to those in the existing literature. In the special case where $g = 0$ in (2.1), (2.9) reduces to;

$$\|DA(x)^\top A(x)\| \ge \nu\|A(x)\|. \tag{2.12}$$

[3]If necessary, to ensure that $\rho < \infty$, one can add a small factor of $\|x\|^2$ to $\mathscr{L}_\beta$ in (1.4). Then it is easy to verify that the iterates of Algorithm 1 remain bounded, provided that the initial penalty weight β_0 is large enough, $\sup_x \|\nabla f(x)\|/\|x\| < \infty$, $\sup_x \|A(x)\| < \infty$, and $\sup_x \|DA(x)\| < \infty$.

Polyak-Lojasiewicz (PL) condition [KNS16]. Consider the problem with $\lambda_{\tilde{f}}$-smooth objective,

$$\min_{x\in\mathbb{R}^d} \tilde{f}(x).$$

$\tilde{f}(x)$ satisfies the PL inequality if the following holds for some $\mu > 0$,

$$\frac{1}{2}\|\nabla \tilde{f}(x)\|^2 \geq \mu(\tilde{f}(x) - \tilde{f}^*), \quad \forall x \qquad \text{(PL inequality)}$$

This inequality implies that gradient is growing faster than a quadratic as we move away from the optimal. Assuming that the feasible set $\{x : A(x) = 0\}$ is non-empty, it is easy to verify that 2.12 is equivalent to the PL condition for minimizing $\tilde{f}(x) = \frac{1}{2}\|A(x)\|^2$ with $\nu = \sqrt{2\mu}$ [KNS16].

PL condition itself is a special case of Kurdyka-Lojasiewicz with $\theta = 1/2$, see [XY17, Definition 1.1]. When $g = 0$, it is also easy to see that (2.9) is weaker than the Mangasarian-Fromovitz (MF) condition in nonlinear optimization [BST18, Assumption 1]. Moreover, when g is the indicator on a convex set, (2.9) is a consequence of the *basic constraint qualification* in [Roc93], which itself generalizes the MF condition to the case when g is an indicator function of a convex set.

We may think of (2.9) as a local condition, which should hold within a neighborhood of the constraint set $\{x : A(x) = 0\}$ rather than everywhere in $\mathbb{R}^d$. Indeed, the iteration count k appears in (2.9) to reflect this local nature of the condition. Similar kind of arguments on the regularity condition also appear in [BST18]. There is also a constant complexity algorithm in [BST18] to reach so-called "information zone", which supplements Theorem 2.4.1.

Penalty method.A classical algorithm to solve (2.1) is the penalty method, which is characterized by the absence of the dual variable ($y = 0$) in (1.4). Indeed, ALM can be interpreted as an adaptive penalty or smoothing method with a variable center determined by the dual variable. It is worth noting that, with the same proof technique, one can establish the same convergence rate of Theorem 2.4.1 for the penalty method. However, while both methods have the same convergence rate in theory, we ignore the uncompetitive penalty method since it is significantly outperformed by iALM in practice.

Computational complexity. Theorem 2.4.1 specifies the number of (outer) iterations that Algorithm 1 requires to reach a near-stationary point of problem (1.4) with a prescribed precision and, in particular, specifies the number of calls made to the solver in Step 2. In this sense, Theorem 2.4.1 does not fully capture the computational complexity of Algorithm 1, as it does not take into account the computational cost of the solver in Step 2.

To better understand the total iteration complexity of Algorithm 1, we consider two scenarios in the following. In the first scenario, we take the solver in Step 2 to be the Accelerated Proximal Gradient Method (APGM), a well-known first-order algorithm [GL16]. In the second scenario, we will use the second-order trust region method developed in [CGT12]. We have the following

two corollaries showing the total complexity of our algorithm to reach first and second-order stationary points. Section 2.7 contains the proofs and more detailed discussion for the complexity results.

Corollary 2.4.2 (First-order optimality)**.** *For $b > 1$, let $\beta_k = b^k$ for every k. If we use APGM from [GL16] for Step 2 of Algorithm 1, the algorithm finds an (ϵ_f, β_k) first-order stationary point of* (1.3)*, after T calls to the first-order oracle, where*

$$T = \mathcal{O}\left(\frac{Q^3\rho^2}{\epsilon^4}\log_b\left(\frac{Q}{\epsilon}\right)\right) = \tilde{\mathcal{O}}\left(\frac{Q^3\rho^2}{\epsilon^4}\right). \tag{2.13}$$

For Algorithm 1 to reach a near-stationary point with an accuracy of ϵ_f in the sense of (1.7) and with the lowest computational cost, we therefore need to perform only one iteration of Algorithm 1, with β_1 specified as a function of ϵ_f by (2.10) in Theorem 2.4.1. In general, however, the constants in (2.10) are unknown and this approach is thus not feasible. Instead, the homotopy approach taken by Algorithm 1 ensures achieving the desired accuracy by gradually increasing the penalty weight. This homotopy approach increases the computational cost of Algorithm 1 only by a factor logarithmic in the ϵ_f, as detailed in the proof of Corollary 2.4.2.

Corollary 2.4.3 (Second-order optimality)**.** *For $b > 1$, let $\beta_k = b^k$ for every k. We assume that*

$$\mathscr{L}_\beta(x_1, y) - \min_x \mathscr{L}_\beta(x, y) \le L_u, \qquad \forall \beta. \tag{2.14}$$

If we use the trust region method from [CGT12] for Step 2 of Algorithm 1, the algorithm finds an ϵ-second-order stationary point of (1.3) *in T calls to the second-order oracle where*

$$T = \mathcal{O}\left(\frac{L_u Q'^5}{\epsilon^5}\log_b\left(\frac{Q'}{\epsilon}\right)\right) = \tilde{\mathcal{O}}\left(\frac{L_u Q'^5}{\epsilon^5}\right). \tag{2.15}$$

Remark. These complexity results for first and second-order are stationarity with respect to (1.4). We note that second order complexity result matches [CGT18] and [BGM$^+$16]. However, the stationarity criteria and the definition of dual variable in these papers differ from ours. We include more discussion on this in the Section 2.7.1.

Effect of β_k in 2.9. We consider two cases, when g is the indicator of a convex set (or 0), the subdifferential set will be a cone (or 0), thus β_k will not have an effect. On the other hand, when g is a convex and Lipschitz contiunous function defined on the whole space, subdifferential set will be bounded [Roc70, Theorem 23.4]. This will introduce an error term in 2.9 that is of the order $(1/\beta_k)$. One can see that b^k choice for β_k causes a linear decrease in this error term. In fact, all the examples in this paper fall into the first case.

2.5 Related Work

ALM has a long history in the optimization literature, dating back to [Hes69, Pow69]. In the special case of (2.1) with a convex function f and a linear operator A, standard, inexact, and linearized versions of ALM have been extensively studied [LM16, NNTD14, TDAFC18, Xu17b].

Classical works on ALM focused on the general template of (2.1) with non-convex f and nonlinear A, with arguably stronger assumptions and required exact solutions to the subproblems of the form (1.5), which appear in Step 2 of Algorithm 1, see for instance [Ber14].

A similar analysis was conducted in [FS12] for the general template of (2.1). The authors considered inexact ALM and proved convergence rates for the outer iterates, under specific assumptions on the initialization of the dual variable. However, in contrast, the authors did not analyze how to solve the subproblems inexactly and did not provide total complexity results with verifiable conditions.

Problem (2.1) with similar assumptions to us is also studied in [BGM+16] and [CGT18] for first-order and second-order stationarity, respectively, with explicit iteration complexity analysis. As we have mentioned in Section 2.4, our second order iteration complexity result matches these theoretical algorithms with a simpler algorithm and a simpler analysis. In addition, these algorithms require setting final accuracies since they utilize this information in the algorithm while our Algorithm 1 does not set accuracies a priori.

[CGT11] also considers the same template (2.1) for first-order stationarity with a penalty-type method instead of ALM. Even though the authors show $\mathcal{O}(1/\epsilon^2)$ complexity, this result is obtained by assuming that the penalty parameter remains bounded. We note that such an assumption can also be used to improve our complexity results to match theirs.

[BST18] studies the general template (2.1) with specific assumptions involving local error bound conditions for the (2.1). These conditions are studied in detail in [BNPS17], but their validity for general SDPs (2.2) has never been established. This work also lacks the total iteration complexity analysis presented here.

Another work [CMV18] focused on solving (2.1) by adapting the primal-dual method of Chambolle and Pock [CP11]. The authors proved the convergence of the method and provided convergence rate by imposing error bound conditions on the objective function that do not hold for standard SDPs.

[BM03b, BM05] is the first work that proposes the splitting $X = UU^\top$ for solving SDPs of the form (2.2). Following these works, the literature on Burer-Monteiro (BM) splitting for the large part focused on using ALM for solving the reformulated problem (2.3). However, this proposal has a few drawbacks: First, it requires exact solutions in Step 2 of Algorithm 1 in theory, which in practice is replaced with inexact solutions. Second, their results only establish convergence without providing the rates. In this sense, our work provides a theoretical understanding of the

BM splitting with inexact solutions to Step 2 of Algorithm 1 and complete iteration complexities.

[BKS16, PKB$^+$16] are among the earliest efforts to show convergence rates for BM splitting, focusing on the special case of SDPs without any linear constraints. For these specific problems, they prove the convergence of gradient descent to global optima with convergence rates, assuming favorable initialization. These results, however, do not apply to general SDPs of the form (2.2) where the difficulty arises due to the linear constraints.

Another popular method for solving SDPs are due to [BMAS14, BAC16, BVB16], focusing on the case where the constraints in (2.1) can be written as a Riemannian manifold after BM splitting. In this case, the authors apply the Riemannian gradient descent and Riemannian trust region methods for obtaining first- and second-order stationary points, respectively. They obtain $\mathcal{O}(1/\epsilon^2)$ complexity for finding first-order stationary points and $\mathcal{O}(1/\epsilon^3)$ complexity for finding second-order stationary points.

While these complexities appear better than ours, the smooth manifold requirement in these works is indeed restrictive. In particular, this requirement holds for max-cut and generalized eigenvalue problems, but it is not satisfied for other important SDPs such as quadratic programming (QAP), optimal power flow and clustering with general affine constraints. In addition, as noted in [BAC16], per iteration cost of their method for max-cut problem is an astronomical $\mathcal{O}(d^6)$.

[Cif21] extends the approach of [BAC16] to arbitrary semidefinite programs beyond smooth manifold assumption, possibly involving inequalities or multiple semidefinite constraints. This work establishes the relationship between the critical points of factorized problem (2.3) and the original formulation (2.2) for general SDPs satisfying Pataki bound [Pat98, Bar95], deriving similar guarantees as [BAC16]. Their Theorem 1 which formalizes this connection does not require any regularity condition. Nonetheless, the work [Cif21] is only limited to establishing the theoretical connections and does not extend to providing a practical algorithm for solving these problems.

Lastly, there also exists a line of work for solving SDPs in their original convex formulation, in a storage efficient way [Nes09, YDC15, YFLC18]. These works have global optimality guarantees by their virtue of directly solving the convex formulation. On the downside, these works require the use of eigenvalue routines and exhibit significantly slower convergence as compared to non-convex approaches [Jag13].

2.6 Numerical Evidence

We first begin with a caveat: It is known that quasi-Newton methods, such as BFGS and lBFGS, might not converge for non-convex problems [Dai02, Mas04]. For this reason, we have used the trust region method as the second-order solver in our analysis in Section 2.4, which is well-studied for non-convex problems [CGT12]. Empirically, however, BFGS and lBGFS are extremely successful and we have therefore opted for those solvers in this section since the

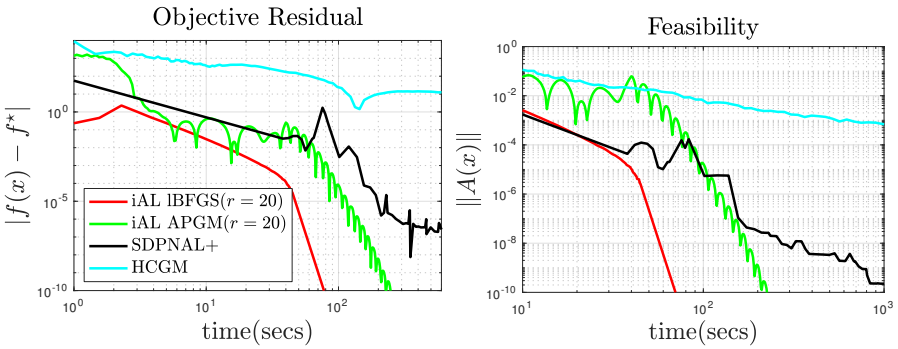

Figure 2.1 – Clustering running time comparison.

subroutine does not affect Theorem 2.4.1 as long as the subsolver performs well in practice.

2.6.1 Clustering

Given data points $\{z_i\}_{i=1}^n$, the entries of the corresponding Euclidean distance matrix $D \in \mathbb{R}^{n\times n}$ are $D_{i,j} = \|z_i - z_j\|^2$. Clustering is then the problem of finding a co-association matrix $Y \in \mathbb{R}^{n\times n}$ such that $Y_{ij} = 1$ if points z_i and z_j are within the same cluster and $Y_{ij} = 0$ otherwise. In [PW07], the authors provide a SDP relaxation of the clustering problem, specified as

$$\min_{Y\in\mathbb{R}^{nxn}} \operatorname{tr}(DY) \quad \text{s.t.} \quad Y\mathbf{1} = \mathbf{1},\ \operatorname{tr}(Y) = s,\ Y \geq 0,\ Y \geq 0, \tag{2.16}$$

where s is the number of clusters and Y is both positive semidefinite and has nonnegative entries. Standard SDP solvers do not scale well with the number of data points n, since they often require projection onto the semidefinite cone with the complexity of $\mathcal{O}(n^3)$. We instead use the BM factorization to solve (2.16), sacrificing convexity to reduce the computational complexity. More specifically, we solve the program

$$\min_{V\in\mathbb{R}^{n\times r}} \operatorname{tr}(DVV^\top) \quad \text{s.t.} \quad VV^\top\mathbf{1} = \mathbf{1},\ \|V\|_F^2 \leq s,\ V \geq 0, \tag{2.17}$$

where $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. Note that $Y \geq 0$ in (2.16) is replaced above by the much stronger but easier-to-enforce constraint $V \geq 0$ in (2.17), see [KSP07] for the reasoning behind this relaxation. Now, we can cast (2.17) as an instance of (2.1). Indeed, for every $i \leq n$, let $x_i \in \mathbb{R}^r$ denote the ith row of V. We next form $x \in \mathbb{R}^d$ with $d = nr$ by expanding the factorized variable V, namely, $x := [x_1^\top, \cdots, x_n^\top]^\top \in \mathbb{R}^d$, and then set

$$f(x) = \sum_{i,j=1}^n D_{i,j}\langle x_i, x_j\rangle, \qquad g = \delta_C, \qquad A(x) = [x_1^\top \sum_{j=1}^n x_j - 1, \cdots, x_n^\top \sum_{j=1}^n x_j - 1]^\top,$$

where C is the intersection of the positive orthant in $\mathbb{R}^d$ with the Euclidean ball of radius $\sqrt{s}$. In Section 2.10, we verify that Theorem 2.4.1 applies to (2.1) with f, g, A specified above.

In our simulations, we use two different solvers for Step 2 of Algorithm 1, namely, APGM and lBFGS. APGM is a solver for non-convex problems of the form (1.5) with convergence guarantees to first-order stationarity, as discussed in Section 2.4. lBFGS is a limited-memory version of BFGS algorithm in [Fle13] that approximately leverages the second-order information of the problem. We compare our approach against the following convex methods:

- HCGM: Homotopy-based Conditional Gradient Method in [YFLC18] which directly solves (2.16).

- SDPNAL+: A second-order augmented Lagrangian method for solving SDP's with non-negativity constraints [YST15].

As for the dataset, our experimental setup is similar to that described by [MVW17]. We use the publicly-available fashion-MNIST data in [XRV17], which is released as a possible replacement for the MNIST handwritten digits. Each data point is a 28×28 gray-scale image, associated with a label from ten classes, labeled from 0 to 9. First, we extract the meaningful features from this dataset using a simple two-layer neural network with a sigmoid activation function. Then, we apply this neural network to 1000 test samples from the same dataset, which gives us a vector of length 10 for each data point, where each entry represents the posterior probability for each class. Then, we form the ℓ_2 distance matrix D from these probability vectors. The solution rank for the template (2.16) is known and it is equal to number of clusters k [KSP07, Theorem 1]. As discussed in [TSC18], setting rank $r > k$ leads more accurate reconstruction in expense of speed. Therefore, we set the rank to 20. For iAL lBFGS, we used $\beta_1 = 1$ and $\sigma_1 = 10$ as the initial penalty weight and dual step size, respectively. For HCGM, we used $\beta_0 = 1$ as the initial smoothness parameter. We have run SDPNAL+ solver with 10^{-12} tolerance. The results are depicted in Figure 2.1. We implemented 3 algorithms on MATLAB and used the software package for SDPNAL+ which contains mex files. It is predictable that the performance of our non-convex approach would even improve by using mex files.

2.6.2 Additional demonstrations

We provide several additional experiments in Section 2.11. Section 2.11.1 discusses a novel non-convex relaxation of the standard basis pursuit template which performs comparable to the state of the art convex solvers. In Section 2.11.2, we provide fast numerical solutions to the generalized eigenvalue problem. In Section 2.11.3, we give a contemporary application example that our template applies, namely, denoising with generative adversarial networks. Finally, we provide improved bounds for sparse quadratic assignment problem instances in Section 2.11.4.

2.7 Complexity Results

2.7.1 First-Order Optimality

Let us first consider the case where the solver in Step 2 is is the first-order algorithm APGM, described in detail in [GL16]. At a high level, APGM makes use of $\nabla_x \mathscr{L}_\beta(x, y)$ in (1.4), the proximal operator prox_g, and the classical Nesterov acceleration [Nes83] to reach first-order stationarity for the subproblem in (1.5). Suppose that $g = \delta_{\mathscr{X}}$ is the indicator function on a bounded convex set $\mathscr{X} \subset \mathbb{R}^d$ and let

$$\rho = \max_{x \in \mathscr{X}} \|x\|, \tag{2.18}$$

be the radius of a ball centered at the origin that includes $\mathscr{X}$. Then, adapting the results in [GL16] to our setup, APGM reaches x_k in Step 2 of Algorithm 1 after

$$\mathscr{O}\left(\frac{\lambda_{\beta_k}^2 \rho^2}{\epsilon_{k+1}^2}\right) \tag{2.19}$$

(inner) iterations, where λ_{β_k} denotes the Lipschitz constant of $\nabla_x \mathscr{L}_{\beta_k}(x, y)$, bounded in (2.6). For the clarity of the presentation, we have used a looser bound in (2.19) compared to [GL16]. Using (2.19), we derive the following corollary, describing the total iteration complexity of Algorithm 1 in terms of the number calls made to the first-order oracle in APGM.

Corollary 2.7.1. *For $b > 1$, let $\beta_k = b^k$ for every k. If we use APGM from [GL16] for Step 2 of Algorithm 1, the algorithm finds an (ϵ_f, β_k) first-order stationary point, after T calls to the first-order oracle, where*

$$T = \mathscr{O}\left(\frac{Q^3 \rho^2}{\epsilon^4} \log_b\left(\frac{Q}{\epsilon}\right)\right) = \tilde{\mathscr{O}}\left(\frac{Q^3 \rho^2}{\epsilon^4}\right). \tag{2.20}$$

Proof. Let K denote the number of (outer) iterations of Algorithm 1 and let ϵ_f denote the desired accuracy of Algorithm 1, see (1.7). Recalling Theorem 2.4.1, we can then write that

$$\epsilon_f = \frac{Q}{\beta_K}, \tag{2.21}$$

or, equivalently, $\beta_K = Q/\epsilon_f$. We now count the number of total (inner) iterations T of Algorithm 1 to reach the accuracy ϵ_f. From (2.6) and for sufficiently large k, recall that $\lambda_{\beta_k} \leq \lambda'' \beta_k$ is the smoothness parameter of the augmented Lagrangian. Then, from (2.19) ad by summing over the

outer iterations, we bound the total number of (inner) iterations of Algorithm 1 as

$$\begin{aligned} T &= \sum_{k=1}^{K} \mathcal{O}\left(\frac{\lambda_{\beta_{k-1}}^2 \rho^2}{\epsilon_k^2}\right) \\ &= \sum_{k=1}^{K} \mathcal{O}\left(\beta_{k-1}^4 \rho^2\right) \qquad \text{(Step 1 of Algorithm 1)} \\ &\leq \mathcal{O}\left(K \beta_{K-1}^4 \rho^2\right) \qquad \left(\{\beta_k\}_k \text{ is increasing}\right) \\ &\leq \mathcal{O}\left(\frac{K Q^3 \rho^2}{\epsilon_f^4}\right). \qquad \text{(see (2.21))} \end{aligned} \tag{2.22}$$

In addition, if we specify $\beta_k = b^k$ for all k, we can further refine T. Indeed,

$$\beta_K = b^K \implies K = \log_b\left(\frac{Q}{\epsilon_f}\right), \tag{2.23}$$

which, after substituting into (2.22) gives the final bound in Corollary 2.4.2. □

2.7.2 Second-Order Optimality

Let us now consider the second-order optimality case where the solver in Step 2 is the the trust region method developed in [CGT12]. Trust region method minimizes a quadratic approximation of the function within a dynamically updated trust-region radius. Second-order trust region method that we consider in this section makes use of Hessian (or an approximation of Hessian) of the augmented Lagrangian in addition to first order oracles.

As shown in [NLR18], finding approximate second-order stationary points of convex-constrained problems is in general NP-hard. For this reason, we focus in this section on the special case of (2.1) with $g = 0$.

Let us compute the total computational complexity of Algorithm 1 with the trust region method in Step 2, in terms of the number of calls made to the second-order oracle. By adapting the result in [CGT12] to our setup, we find that the number of (inner) iterations required in Step 2 of Algorithm 1 to produce x_{k+1} is

$$\mathcal{O}\left(\frac{\lambda_{\beta_k,H}^2 (\mathscr{L}_{\beta_k}(x_1, y) - \min_x \mathscr{L}_{\beta_k}(x, y))}{\epsilon_k^3}\right), \tag{2.24}$$

where $\lambda_{\beta,H}$ is the Lipschitz constant of the Hessian of the augmented Lagrangian, which is of the order of β, as can be proven similar to Lemma 2.2.1 and x_1 is the initial iterate of the given outer loop. In [CGT12], the term $\mathscr{L}_\beta(x_1, y) - \min_x \mathscr{L}_\beta(x, y)$ is bounded by a constant independent of ϵ. We assume a uniform bound for this quantity for every β_k, instead of for one value of β_k as in [CGT12]. Using (2.24) and Theorem 2.4.1, we arrive at the following:

Corollary 2.7.2. *For $b > 1$, let $\beta_k = b^k$ for every k. We assume that*

$$\mathscr{L}_\beta(x_1, y) - \min_x \mathscr{L}_\beta(x, y) \le L_u, \qquad \forall \beta. \tag{2.25}$$

If we use the trust region method from [CGT12] for Step 2 of Algorithm 1, the algorithm finds an ϵ-second-order stationary point of (2.1) in T calls to the second-order oracle where

$$T = \mathscr{O}\left(\frac{L_u Q'^5}{\epsilon^5} \log_b\left(\frac{Q'}{\epsilon}\right)\right) = \widetilde{\mathscr{O}}\left(\frac{L_u Q'^5}{\epsilon^5}\right). \tag{2.26}$$

Before closing this section, we note that the remark after Corollary 2.4.2 applies here as well.

2.7.3 Approximate optimality of (2.1).

Corollary 2.4.2 establishes the iteration complexity of Algorithm 1 to reach approximate first-order stationarity for the equivalent formulation of (2.1) presented in (1.3). Unlike the exact case, approximate first-order stationarity in (1.3) does not immediately lend itself to approximate stationarity in (2.1), and the study of approximate stationarity for the penalized problem (special case of our setting with dual variable set to 0) has also precedent in [BBJN18]. For a precedent in convex optimization for relating the convergence in augmented Lagrangian to the constrained problem using duality, see [TDFC18]. For the second-order case, it is in general not possible to establish approximate second-order optimality for (1.3) from Corollary 2.4.3, with the exception of linear constraints. [NLR18] provides an hardness result by showing that checking an approximate second-order stationarity is NP-hard.

2.8 Proof of Theorem 2.4.1

For every $k \ge 2$, recall from (1.4) and Step 2 of Algorithm 1 that x_k satisfies

$$\begin{aligned} &\operatorname{dist}(-\nabla f(x_k) - DA(x_k)^\top y_{k-1} - \beta_{k-1} DA(x_k)^\top A(x_k), \partial g(x_k)) \\ &= \operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1}), \partial g(x_k)) \le \epsilon_k. \end{aligned} \tag{2.27}$$

With an application of the triangle inequality, it follows that

$$\operatorname{dist}(-\beta_{k-1} DA(x_k)^\top A(x_k), \partial g(x_k)) \le \|\nabla f(x_k)\| + \|DA(x_k)^\top y_{k-1}\| + \epsilon_k, \tag{2.28}$$

which in turn implies that

$$\begin{aligned} \operatorname{dist}(-DA(x_k)^\top A(x_k), \partial g(x_k)/\beta_{k-1}) &\le \frac{\|\nabla f(x_k)\|}{\beta_{k-1}} + \frac{\|DA(x_k)^\top y_{k-1}\|}{\beta_{k-1}} + \frac{\epsilon_k}{\beta_{k-1}} \\ &\le \frac{\lambda'_f + \lambda'_A \|y_{k-1}\| + \epsilon_k}{\beta_{k-1}}, \end{aligned} \tag{2.29}$$

where λ'_f, λ'_A were defined in (2.8). We next translate (2.29) into a bound on the feasibility gap $\|A(x_k)\|$. Using the regularity condition (2.9), the left-hand side of (2.29) can be bounded below as

$$\operatorname{dist}(-DA(x_k)^\top A(x_k), \partial g(x_k)/\beta_{k-1}) \geq \nu \|A(x_k)\|. \qquad \text{(see (2.9))} \tag{2.30}$$

By substituting (2.30) back into (2.29), we find that

$$\|A(x_k)\| \leq \frac{\lambda'_f + \lambda'_A \|y_{k-1}\| + \epsilon_k}{\nu \beta_{k-1}}. \tag{2.31}$$

In words, the feasibility gap is directly controlled by the dual sequence $\{y_k\}_k$. We next establish that the dual sequence is bounded. Indeed, for every $k \in K$, note that

$$\begin{aligned}
\|y_k\| &= \|y_0 + \sum_{i=1}^{k} \sigma_i A(x_i)\| \quad \text{(Step 5 of Algorithm 3)} \\
&\leq \|y_0\| + \sum_{i=1}^{k} \sigma_i \|A(x_i)\| \qquad \text{(triangle inequality)} \\
&\leq \|y_0\| + \sum_{i=1}^{k} \frac{\|A(x_1)\| \log^2 2}{k \log^2(k+1)} \quad \text{(Step 4)} \\
&\leq \|y_0\| + c\|A(x_1)\| \log^2 2 =: y_{\max},
\end{aligned} \tag{2.32}$$

where

$$c \geq \sum_{i=1}^{\infty} \frac{1}{k \log^2(k+1)}. \tag{2.33}$$

Substituting (2.32) back into (2.31), we reach

$$\|A(x_k)\| \leq \frac{\lambda'_f + \lambda'_A y_{\max} + \epsilon_k}{\nu \beta_{k-1}} \leq \frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu \beta_{k-1}}, \tag{2.34}$$

where the second inequality above holds if k_0 is large enough, which would in turn guarantees that $\epsilon_k = 1/\beta_{k-1}$ is sufficiently small since $\{\beta_k\}_k$ is increasing and unbounded. It remains to control the first term in (1.8). To that end, after recalling Step 2 of Algorithm 1 and applying the triangle inequality, we can write that

$$\begin{aligned}
\operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) \leq \operatorname{dist}(&-\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1}), \partial g(x_k)) \\
&+ \|\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k) - \nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1})\|.
\end{aligned} \tag{2.35}$$

The first term on the right-hand side above is bounded by ϵ_k, by Step 5 of Algorithm 1. For the second term on the right-hand side of (2.35), we write that

$$\begin{aligned}\|\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k) - \nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1})\| &= \|DA(x_k)^\top (y_k - y_{k-1})\| \qquad \text{(see (1.4))} \\ &\leq \lambda'_A \|y_k - y_{k-1}\| \qquad \text{(see (2.8))} \\ &= \lambda'_A \sigma_k \|A(x_k)\| \qquad \text{(see Step 5 of Algorithm 1)} \\ &\leq \frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}). \qquad \text{(see (2.34))} \end{aligned} \tag{2.36}$$

By combining (2.35,2.36), we find that

$$\operatorname{dist}(\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) \leq \frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k. \tag{2.37}$$

By combining (2.34,2.37), we find that

$$\begin{aligned}&\operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) + \|A(x_k)\| \\ &\leq \left(\frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k \right) \\ &\qquad + 2 \left(\frac{\lambda'_f + \lambda'_A y_{\max}}{\nu \beta_{k-1}} \right). \end{aligned} \tag{2.38}$$

Applying $\sigma_k \leq \sigma_1$, we find that

$$\begin{aligned}&\operatorname{dist}(-\nabla_x \mathscr{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) + \|A(x_k)\| \\ &\leq \frac{2\lambda'_A \sigma_1 + 2}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k. \end{aligned} \tag{2.39}$$

For the second part of the theorem, we use the Weyl's inequality and Step 5 of Algorithm 1 to write

$$\begin{aligned}\lambda_{\min}(\nabla_{xx} \mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1})) &\geq \lambda_{\min}(\nabla_{xx} \mathscr{L}_{\beta_{k-1}}(x_k, y_k)) \\ &\quad - \sigma_k \| \sum_{i=1}^m A_i(x_k) \nabla^2 A_i(x_k) \|. \end{aligned} \tag{2.40}$$

The first term on the right-hand side is lower bounded by $-\epsilon_{k-1}$ by Step 2 of Algorithm 1. We next bound the second term on the right-hand side above as

$$\begin{aligned}\sigma_k \| \sum_{i=1}^m A_i(x_k) \nabla^2 A_i(x_k) \| &\leq \sigma_k \sqrt{m} \max_i \|A_i(x_k)\| \|\nabla^2 A_i(x_k)\| \\ &\leq \sigma_k \sqrt{m} \lambda_A \frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu \beta_{k-1}}, \end{aligned}$$

where the last inequality is due to (2.4,2.34). Plugging into (2.40) gives

$$\begin{aligned}&\lambda_{\min}(\nabla_{xx}\mathscr{L}_{\beta_{k-1}}(x_k, y_{k-1}))\\ &\geq -\epsilon_{k-1} - \sigma_k\sqrt{m}\lambda_A\frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu\beta_{k-1}},\end{aligned}$$

which completes the proof of Theorem 2.4.1.

2.9 Proof of Lemma 2.2.1

Proof. Note that

$$\mathscr{L}_\beta(x, y) = f(x) + \sum_{i=1}^m y_i A_i(x) + \frac{\beta}{2}\sum_{i=1}^m (A_i(x))^2, \tag{2.41}$$

which implies that

$$\begin{aligned}&\nabla_x\mathscr{L}_\beta(x, y)\\ &= \nabla f(x) + \sum_{i=1}^m y_i\nabla A_i(x) + \frac{\beta}{2}\sum_{i=1}^m A_i(x)\nabla A_i(x)\\ &= \nabla f(x) + DA(x)^\top y + \beta DA(x)^\top A(x),\end{aligned} \tag{2.42}$$

where $DA(x)$ is the Jacobian of A at x. By taking another derivative with respect to x, we reach

$$\begin{aligned}\nabla_x^2\mathscr{L}_\beta(x, y) = \nabla^2 f(x) + \sum_{i=1}^m \big(y_i + \beta A_i(x)\big)\nabla^2 A_i(x)\\ + \beta\sum_{i=1}^m \nabla A_i(x)\nabla A_i(x)^\top.\end{aligned} \tag{2.43}$$

It follows that

$$\begin{aligned}&\|\nabla_x^2\mathscr{L}_\beta(x, y)\|\\ &\leq \|\nabla^2 f(x)\| + \max_i \|\nabla^2 A_i(x)\|\left(\|y\|_1 + \beta\|A(x)\|_1\right)\\ &\quad + \beta\sum_{i=1}^m \|\nabla A_i(x)\|^2\\ &\leq \lambda_h + \sqrt{m}\lambda_A\left(\|y\| + \beta\|A(x)\|\right) + \beta\|DA(x)\|_F^2.\end{aligned} \tag{2.44}$$

For every x such that $\|x\| \leq \rho$ and $\|A(x)\| \leq \rho$, we conclude that

$$\|\nabla_x^2\mathscr{L}_\beta(x, y)\| \leq \lambda_f + \sqrt{m}\lambda_A\left(\|y\| + \beta\rho'\right) + \beta\max_{\|x\|\leq\rho}\|DA(x)\|_F^2, \tag{2.45}$$

which completes the proof of Lemma 2.2.1. □

2.10 Clustering

We only verify the condition in (2.9) here. Note that

$$A(x) = VV^\top \mathbf{1} - \mathbf{1}, \tag{2.46}$$

$$\begin{aligned} DA(x) &= \begin{bmatrix} w_{1,1}x_1^\top & \cdots & w_{1,n}x_1^\top \\ \vdots & & \\ w_{n,1}x_n^\top & \cdots & w_{n,n}1x_n^\top \end{bmatrix} \\ &= \begin{bmatrix} V & \cdots & V \end{bmatrix} + \begin{bmatrix} x_1^\top & & \\ & \ddots & \\ & & x_n^\top \end{bmatrix}, \end{aligned} \tag{2.47}$$

where $w_{i.i} = 2$ and $w_{i,j} = 1$ for $i \neq j$. In the last line above, n copies of V appear and the last matrix above is block-diagonal. For x_k, define V_k accordingly and let $x_{k,i}$ be the ith row of V_k. Consequently,

$$\begin{aligned} DA(x_k)^\top A(x_k) = &\begin{bmatrix} (V_k^\top V_k - I_n)V_k^\top \mathbf{1} \\ \vdots \\ (V_k^\top V_k - I_n)V_k^\top \mathbf{1} \end{bmatrix} \\ &+ \begin{bmatrix} x_{k,1}(V_k V_k^\top \mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_{k,n}(V_k V_k^\top \mathbf{1} - \mathbf{1})_n \end{bmatrix}, \end{aligned} \tag{2.48}$$

where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. Let us make a number of simplifying assumptions. First, we assume that $\|x_k\| < \sqrt{s}$ (which can be enforced in the iterates by replacing C with $(1-\epsilon)C$ for a small positive ϵ in the subproblems). Under this assumption, it follows that

$$(\partial g(x_k))_i = \begin{cases} 0 & (x_k)_i > 0 \\ \{a : a \leq 0\} & (x_k)_i = 0, \end{cases} \qquad i \leq d. \tag{2.49}$$

Second, we assume that V_k has nearly orthonormal columns, namely, $V_k^\top V_k \approx I_n$. This can also be enforced in each iterate of Algorithm 1 and naturally corresponds to well-separated clusters. While a more fine-tuned argument can remove these assumptions, they will help us simplify the

presentation here. Under these assumptions, the (squared) right-hand side of (2.9) becomes

$$\begin{aligned}
&\operatorname{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right)^2 \\
&= \left\|\left(-DA(x_k)^\top A(x_k)\right)_+\right\|^2 \qquad (a_+ = \max(a,0)) \\
&= \left\| \begin{bmatrix} x_{k,1}(V_k V_k^\top \mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_{k,n}(V_k V_k^\top \mathbf{1} - \mathbf{1})_n \end{bmatrix} \right\|^2 \qquad (x_k \in C \Rightarrow x_k \geq 0) \\
&= \sum_{i=1}^{n} \|x_{k,i}\|^2 (V_k V_k^\top \mathbf{1} - \mathbf{1})_i^2 \\
&\geq \min_i \|x_{k,i}\|^2 \cdot \sum_{i=1}^{n} (V_k V_k^\top \mathbf{1} - \mathbf{1})_i^2 \\
&= \min_i \|x_{k,i}\|^2 \cdot \|V_k V_k^\top \mathbf{1} - \mathbf{1}\|^2. \qquad (2.50)
\end{aligned}$$

Therefore, given a prescribed ν, ensuring $\min_i \|x_{k,i}\| \geq \nu$ guarantees (2.9). When the algorithm is initialized close enough to the constraint set, there is indeed no need to separately enforce (2.50). In practice, often n exceeds the number of true clusters and a more intricate analysis is required to establish (2.9) by restricting the argument to a particular subspace of $\mathbb{R}^n$.

2.11 Additional Experiments

2.11.1 Basis Pursuit

Basis Pursuit (BP) finds sparsest solutions of an under-determined system of linear equations by solving

$$\min_z \|z\|_1 \quad \text{s.t.} \quad Bz = b, \qquad (2.51)$$

where $B \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$. Various primal-dual convex optimization algorithms are available in the literature to solve BP, including [TDAFC18, CP11]. We compare our algorithm against state-of-the-art primal-dual convex methods for solving (2.51), namely, Chambole-Pock [CP11], ASGARD [TDFC18] and ASGARD-DL [TDAFC18].

Here, we take a different approach and cast (2.51) as an instance of (2.1). Note that any $z \in \mathbb{R}^d$ can be decomposed as $z = z^+ - z^-$, where $z^+, z^- \in \mathbb{R}^d$ are the positive and negative parts of z, respectively. Then consider the change of variables $z^+ = u_1^{\circ 2}$ and $z^- = u_2^{\circ 2} \in \mathbb{R}^d$, where $\circ$ denotes element-wise power. Next, we concatenate u_1 and u_2 as $x := [u_1^\top, u_2^\top]^\top \in \mathbb{R}^{2d}$ and define $\overline{B} := [B, -B] \in \mathbb{R}^{n\times 2d}$. Then, (2.51) is equivalent to (2.1) with

$$f(x) = \|x\|^2, \quad g(x) = 0, \quad \text{s.t.} \quad A(x) = \overline{B}x^{\circ 2} - b. \qquad (2.52)$$

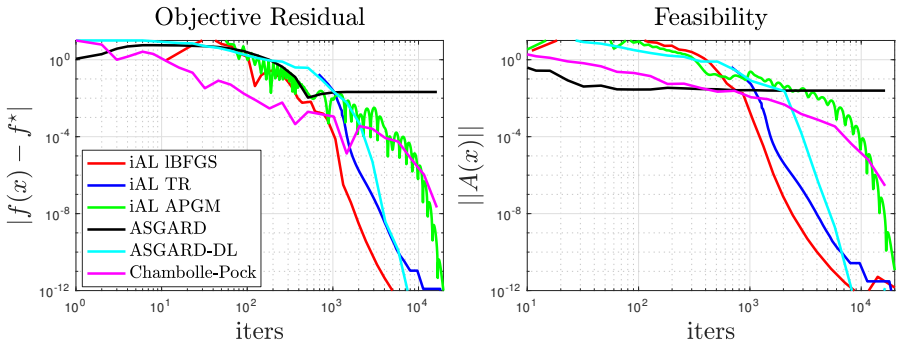

Figure 2.2 – Basis Pursuit

We draw the entries of B independently from a zero-mean and unit-variance Gaussian distribution. For a fixed sparsity level k, the support of $z_* \in \mathbb{R}^d$ and its nonzero amplitudes are also drawn from the standard Gaussian distribution. Then the measurement vector is created as $b = Bz + \epsilon$, where ϵ is the noise vector with entries drawn independently from the zero-mean Gaussian distribution with variance $\sigma^2 = 10^{-6}$.

The results are compiled in Figure 2.2. Clearly, the performance of Algorithm 1 with a second-order solver for BP is comparable to the rest. It is, indeed, interesting to see that these type of non-convex relaxations gives the solution of convex one and first order methods succeed.

Discussion: The true potential of our reformulation is in dealing with more structured norms rather than ℓ_1, where computing the proximal operator is often intractable. One such case is the latent group lasso norm [OJV11], defined as

$$\|z\|_\Omega = \sum_{i=1}^{I} \|z_{\Omega_i}\|,$$

where $\{\Omega_i\}_{i=1}^I$ are (not necessarily disjoint) index sets of $\{1, \cdots, d\}$. Although not studied here, we believe that the non-convex framework presented in this paper can serve to solve more complicated problems, such as the latent group lasso. We leave this research direction for future work.

Condition verification: In the sequel, we verify that Theorem 2.4.1 indeed applies to (2.1) with the above f, A, g. Note that

$$DA(x) = 2\overline{B}\text{diag}(x), \tag{2.53}$$

where $\mathrm{diag}(x) \in \mathbb{R}^{2d\times 2d}$ is the diagonal matrix formed by x. The left-hand side of (2.9) then reads as

$$\begin{aligned}
&\mathrm{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right)\\
&= \mathrm{dist}\left(-DA(x_k)^\top A(x_k), \{0\}\right) \qquad (g \equiv 0)\\
&= \|DA(x_k)^\top A(x_k)\|\\
&= 2\|\mathrm{diag}(x_k)\overline{B}^\top(\overline{B}x_k^{\circ 2} - b)\|. \qquad \text{(see (2.53))}
\end{aligned} \tag{2.54}$$

To bound the last line above, let x_* be a solution of (2.1) and note that $\overline{B}x_*^{\circ 2} = b$ by definition. Let also $z_k, z_* \in \mathbb{R}^d$ denote the vectors corresponding to x_k, x_*. Corresponding to x_k, also define $u_{k,1}, u_{k,2}$ naturally and let $|z_k| = u_{k,1}^{\circ 2} + u_{k,2}^{\circ 2} \in \mathbb{R}^d$ be the vector of amplitudes of z_k. To simplify matters, let us assume also that B is full-rank. We then rewrite the norm in the last line of (2.54) as

$$\begin{aligned}
&\|\mathrm{diag}(x_k)\overline{B}^\top(\overline{B}x_k^{\circ 2} - b)\|^2\\
&= \|\mathrm{diag}(x_k)\overline{B}^\top\overline{B}(x_k^{\circ 2} - x_*^{\circ 2})\|^2 \qquad (\overline{B}x_*^{\circ 2} = b)\\
&= \|\mathrm{diag}(x_k)\overline{B}^\top B(x_k - x_*)\|^2\\
&= \|\mathrm{diag}(u_{k,1})B^\top B(z_k - z_*)\|^2\\
&\qquad + \|\mathrm{diag}(u_{k,2})B^\top B(z_k - z_*)\|^2\\
&= \|\mathrm{diag}(u_{k,1}^{\circ 2} + u_{k,2}^{\circ 2})B^\top B(z_k - z_*)\|^2\\
&= \|\mathrm{diag}(|z_k|)B^\top B(z_k - z_*)\|^2\\
&\geq \eta_n(B\mathrm{diag}(|z_k|))^2\|B(z_k - z_*)\|^2\\
&= \eta_n(B\mathrm{diag}(|z_k|))^2\|Bz_k - b\|^2 \qquad (Bz_* = \overline{B}x_*^{\circ 2} = b)\\
&\geq \min_{|T|=n} \eta_n(B_T)\cdot |z_{k,(n)}|^2\|Bz_k - b\|^2,
\end{aligned} \tag{2.55}$$

where $\eta_n(\cdot)$ returns the nth largest singular value of its argument. In the last line above, B_T is the restriction of B to the columns indexed by T of size n. Moreover, $z_{k,(n)}$ is the nth largest entry of z in magnitude. Given a prescribed ν, (2.9) therefore holds if

$$|z_{k,(n)}| \geq \frac{\nu}{2\sqrt{\min_{|T|=n}\eta_n(B_T)}}, \tag{2.56}$$

for every iteration k. If Algorithm 1 is initialized close enough to the solution z^* and the entries of z^* are sufficiently large in magnitude, there will be no need to directly enforce (2.56).

2.11.2 Generalized Eigenvalue Problem

Generalized eigenvalue problem has extensive applications in machine learning, statistics and data analysis [GJN+16]. The well-known non-convex formulation of the problem is [BVB16]

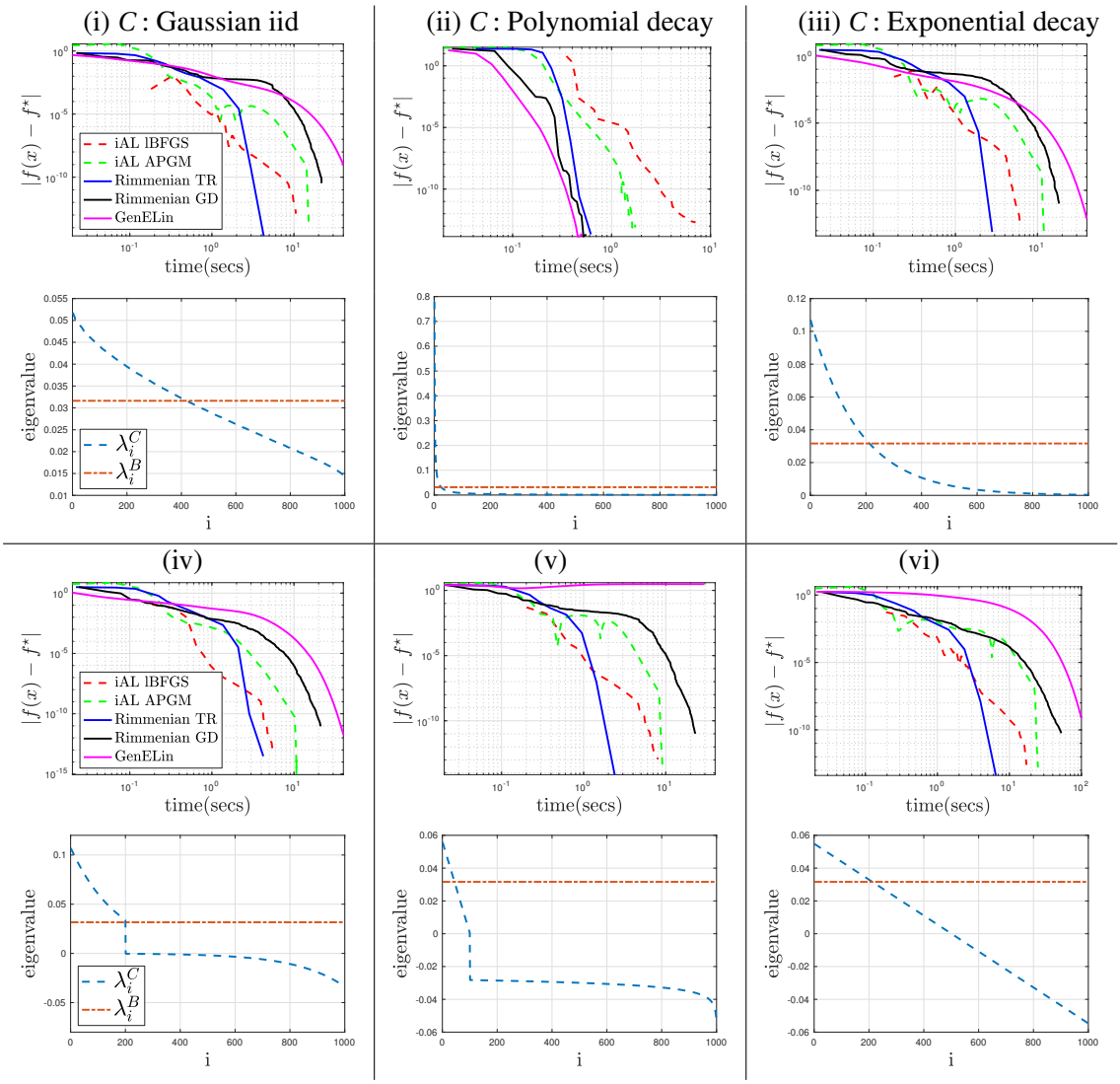

Figure 2.3 – *(Top)* Objective convergence for calculating top generalized eigenvalue and eigenvector of B and C. *(Bottom)* Eigenvalue structure of the matrices. For (i),(ii) and (iii), C is positive semidefinite; for (iv), (v) and (vi), C contains negative eigenvalues. [(i): Generated by taking symmetric part of iid Gaussian matrix. (ii): Generated by randomly rotating $\mathrm{diag}(1^{-p}, 2^{-p}, \cdots, 1000^{-p})(p = 1)$. (iii): Generated by randomly rotating $\mathrm{diag}(10^{-p}, 10^{-2p}, \cdots, 10^{-1000p})(p = 0.0025)$.]

given by

$$\begin{cases} \min\limits_{x \in \mathbb{R}^n} x^\top C x \\ x^\top B x = 1, \end{cases} \tag{2.57}$$

where $B, C \in \mathbb{R}^{n \times n}$ are symmetric matrices and B is positive definite, namely, $B \succ 0$. The generalized eigenvector computation is equivalent to performing principal component analysis (PCA) of C in the norm B. It is also equivalent to computing the top eigenvector of symmetric matrix $S = B^{-1/2} C B^{1/2}$ and multiplying the resulting vector by $B^{-1/2}$. However, for large values

of n, computing $B^{-1/2}$ is extremely expensive. The natural convex SDP relaxation for (2.57) involves lifting $Y = xx^\top$ and removing the non-convex $\text{rank}(Y) = 1$ constraint, namely,

$$\begin{cases} \min\limits_{Y \in \mathbb{R}^{n\times n}} \text{tr}(CY) \\ \text{tr}(BY) = 1, \quad X \succeq 0. \end{cases} \tag{2.58}$$

Here, however, we opt to directly solve (2.57) because it fits into our template with

$$\begin{aligned} f(x) =& x^\top Cx, \quad g(x) = 0, \\ A(x) =& x^\top Bx - 1. \end{aligned} \tag{2.59}$$

We compare our approach against three different methods: manifold based Riemannian gradient descent and Riemannian trust region methods in [BAC16] and the linear system solver in [GJN$^+$16], abbrevated as GenELin. We have used Manopt software package in [BMAS14] for the manifold based methods. For GenELin, we have utilized Matlab's backslash operator as the linear solver. The results are compiled in Figure 2.3.

Condition verification: Here, we verify the regularity condition in (2.9) for problem (2.57). Note that

$$DA(x) = (2Bx)^\top. \tag{2.60}$$

Therefore,

$$\begin{aligned} \text{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right)^2 &= \text{dist}\left(-DA(x_k)^\top A(x_k), \{0\}\right)^2 \qquad (g \equiv 0) \\ &= \|DA(x_k)^\top A(x_k)\|^2 \\ &= \|2Bx_k(x_k^\top Bx_k - 1)\|^2 \qquad \text{(see (2.60))} \\ &= 4(x_k^\top Bx_k - 1)^2 \|Bx_k\|^2 \\ &= 4\|Bx_k\|^2 \|A(x_k)\|^2 \qquad \text{(see (2.59))} \\ &\geq \eta_{\min}(B)^2 \|x_k\|^2 \|A(x_k)\|^2, \end{aligned} \tag{2.61}$$

where $\eta_{\min}(B)$ is the smallest eigenvalue of the positive definite matrix B. Therefore, for a prescribed ν, the regularity condition in (2.9) holds with $\|x_k\| \geq \nu/\eta_{min}$ for every k. If the algorithm is initialized close enough to the constraint set, there will be again no need to directly enforce this latter condition.

2.11.3 ℓ_∞ Denoising with a Generative Prior

The authors of [SKC18, IJA+17] have proposed to project onto the range of a Generative Adversarial network (GAN) [GPM+14], as a way to defend against adversarial examples. For a given noisy observation $x^* + \eta$, they consider a projection in the ℓ_2 norm. We instead propose to use our augmented Lagrangian method to denoise in the ℓ_∞ norm, a much harder task:

$$\begin{aligned} \min_{x,z} \quad & \|x^* + \eta - x\|_\infty \\ \text{s.t.} \quad & x = G(z). \end{aligned} \tag{2.62}$$

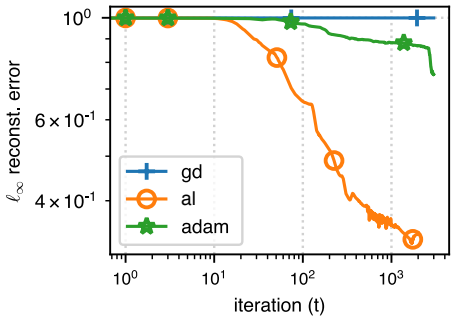

Figure 2.4 – Augmented Lagrangian vs Adam and Gradient descent for ℓ_∞ denoising

We use a pretrained generator for the MNIST dataset, given by a standard deconvolutional neural network architecture [RMC15]. We compare the succesful optimizer Adam [KB14] and gradient descent against our method. Our algorithm involves two forward and one backward pass through the network, as oposed to Adam that requires only one forward/backward pass. For this reason we let our algorithm run for 2000 iterations, and Adam and GD for 3000 iterations. Both Adam and gradient descent generate a sequence of feasible iterates $x_t = G(z_t)$. For this reason we plot the objective evaluated at the point $G(z_t)$ vs iteration count in figure 2.4. Our method successfully minimizes the objective value, while Adam and GD do not.

2.11.4 Quadratic assginment problem

Let K, L be $n \times n$ symmetric metrices. QAP in its simplest form can be written as

$$\max \operatorname{tr}(KPLP), \quad \text{subject to } P \text{ be a permutation matrix} \tag{2.63}$$

A direct approach for solving (2.63) involves a combinatorial search. To get the SDP relaxation of (2.63), we will first lift the QAP to a problem involving a larger matrix. Observe that the objective function takes the form

$$\mathrm{tr}((K \otimes L)(\mathrm{vec}(P)\mathrm{vec}(P^\top))),$$

where $\otimes$ denotes the Kronecker product. Therefore, we can recast (2.63) as

$$\mathrm{tr}((K \otimes L)Y) \text{ subject to } Y = \mathrm{vec}(P)\mathrm{vec}(P^\top), \tag{2.64}$$

where P is a permutation matrix. We can relax the equality constraint in (2.64) to a semidefinite constraint and write it in an equivalent form as

$$X = \begin{bmatrix} 1 & \mathrm{vec}(P)^\top \\ \mathrm{vec}(P) & Y \end{bmatrix} \succeq 0 \text{ for a symmetric} X \in \mathbb{S}^{(n^2+1)\times(n^2+1)}$$

We now introduce the following constraints such that

$$B_k(X) = \mathbf{b_k},\ \mathbf{b_k} \in \mathbb{R}^{m_k} \tag{2.65}$$

to make sure X has a proper structure. Here, B_k is a linear operator on X and the total number of constraints is $m = \sum_k m_k$. Hence, SDP relaxation of the quadratic assignment problem takes the form,

$$\begin{aligned} \max\ & \langle C, X \rangle \\ \text{subject to}\ & P1 = 1,\ 1^\top P = 1,\ P \geq 0 \\ & \mathrm{trace}_1(Y) = I\ \mathrm{trace}_2(Y) = I \\ & \mathrm{vec}(P) = \mathrm{diag}(Y) \\ & \mathrm{trace}(Y) = n \begin{bmatrix} 1 & \mathrm{vec}(P)^\top \\ \mathrm{vec}(P) & Y \end{bmatrix} \succeq 0, \end{aligned} \tag{2.66}$$

where $\mathrm{trace}_1(.)$ and $\mathrm{trace}_2(.)$ are partial traces satisfying,

$$\mathrm{trace}_1(K \otimes L) = \mathrm{trace}(K)L \text{ and } \mathrm{trace}_2(K \otimes L) = K\mathrm{trace}(L)$$

$$\mathrm{trace}_1^*(T) = I \otimes T \text{ and } \mathrm{trace}_2^*(T) = T \otimes I$$

$1st$ set of equalities are due to the fact that permutation matrices are doubly stochastic. $2nd$ set of equalities are to ensure permutation matrices are orthogonal, i.e., $PP^\top = P^\top P = I$. $3rd$ set of equalities are to enforce every individual entry of the permutation matrix takes either 0 or 1, i.e.,

$X_{1,i} = X_{i,i} \ \forall i \in [1, n^2 + 1]$. . Trace constraint in the last line is to bound the problem domain. By concatenating the B_k's in (2.65), we can rewrite (2.66) in standard SDP form as

$$\begin{aligned} \max \ & \langle C, X \rangle \\ \text{subject to } & B(X) = \mathbf{b}, \ \mathbf{b} \in \mathbb{R}^m \\ & \text{trace}(X) = n + 1 \\ & X_{ij} \geq 0, \ \ i, j \ \mathscr{G} \\ & X \succeq 0, \end{aligned} \tag{2.67}$$

where $\mathscr{G}$ represents the index set for which we introduce the nonnegativities. When $\mathscr{G}$ covers the wholes set of indices, we get the best approximation to the original problem. However, it becomes computationally undesirable as the problem dimension increases. Hence, we remove the redundant nonnegativity constraints and enforce it for the indices where Kronecker product between K and L is nonzero.

We penalize the non-negativity constraints and add it to the augmented Lagrangian objective since a projection to the positive orthant approach in the low rank space as we did for the clustering does not work here.

We take [FKS18] as the baseline. This is an SDP based approach for solving QAP problems containing a sparse graph. We compare against the best feasible upper bounds reported in [FKS18] for the given instances. Here, optimality gap is defined as

$$\%\text{Gap} = \frac{|\text{bound} - \text{optimal}|}{\text{optimal}} \times 100$$

We used a (relatively) sparse graph data set from the QAP library. We run our low rank algorithm for different rank values. r_m in each instance corresponds to the smallest integer satisfying the Pataki bound [Pat98, Bar95]. Results are shown in Table 2.1. Primal feasibility values except for the last instance $esc128$ is less than 10^{-5} and we obtained bounds at least as good as the ones reported in [FKS18] for these problems.

For $esc128$, the primal feasibility is $\approx 10^{-1}$, hence, we could not manage to obtain a good optimality gap.

Data	Optimal Value	Optimality Gap (%)					
		Sparse QAP [FKS18]	iAL				
			$r = 10$	$r = 25$	$r = 50$	$r = r_m$	r_m
esc16a	68	8.8	11.8	**0**	**0**	5.9	157
esc16b	292	**0**	**0**	**0**	**0**	**0**	224
esc16c	160	5	5.0	5.0	**2.5**	3.8	177
esc16d	16	12.5	37.5	**0**	**0**	25.0	126
esc16e	28	7.1	7.1	**0**	14.3	7.1	126
esc16g	26	**0**	23.1	7.7	**0**	**0**	126
esc16h	996	**0**	**0**	**0**	**0**	**0**	224
esc16i	14	**0**	**0**	**0**	14.3	**0**	113
esc16j	8	**0**	**0**	**0**	**0**	**0**	106
esc32a	130	93.8	129.2	109.2	104.6	**83.1**	433
esc32b	168	88.1	111.9	92.9	97.6	**69.0**	508
esc32c	642	7.8	15.6	14.0	15.0	**4.0**	552
esc32d	200	21	28.0	28.0	29.0	**17.0**	470
esc32e	2	**0**	**0**	**0**	**0**	**0**	220
esc32g	6	**0**	33.3	**0**	**0**	**0**	234
esc32h	438	18.3	25.1	19.6	25.1	**13.2**	570
esc64a	116	53.4	62.1	51.7	58.6	**34.5**	899
esc128	64	**175**	256.3	193.8	243.8	215.6	2045

Table 2.1 – Comparison between upper bounds on the problems from the QAP library with (relatively) sparse L.

2.12 Bibliographic notes

Figure 2.4 is due to Fabian Latorre. The initial version of Theorem 2.4.1 is mainly due to Armin Eftekhari.

3 Inexact Augmented Lagrangian Method for Solving Factorized SDPs

In this chapter, we consider a canonical nonlinear-constrained non-convex problem with broad applications in machine learning, theoretical computer science, and signal processing. We propose a simple primal-dual splitting scheme that provably converges to a stationary point of the non-convex problem.

We achieve this desideratum via an adaptive and inexact augmented Lagrangian method. The new algorithm features a slow $\mathcal{O}(1/\epsilon^6)$ convergence rate, which it counteracted by its cheap per-iteration complexity. We provide numerical evidence on large-scale machine learning problems, modeled typically via semidefinite relaxations.

3.1 Introduction

We study the following non-convex optimization program:

$$\underset{U\in\mathbb{R}^{n\times r}}{\text{minimize}} \left\{ f(U) : TU = b \quad \text{and} \quad U \in \mathcal{U} \right\} \tag{3.1}$$

where $\mathcal{U}$ is a simple bounded, convex set, such as a Frobenius norm constraint, T is a known nonlinear operator, and b is a known vector in $\mathbb{R}^m$. We assume that f has Lipschitz continuous gradient in U with a Lipschitz constant $L_f \geq 0$.

Countless problems in computer science [KN12, Lov03], machine learning [MNS15, SSGB07], and signal processing [Sin11, SS11, CKS15] naturally fall under this template, including but not limited to maximum cut, clustering and community detection, as well as phase retrieval from magnitude measurements.

An example of our template in (3.1) is semidefinite programming which provides a powerful relaxation approach to the above problems

$$\underset{X\in\mathbb{R}^{n\times n}}{\text{minimize}} \left\{ g(X) : AX = b \quad \text{and} \quad X \in \mathcal{X} \right\} \tag{P}$$

where $\mathscr{X}$ is a simple set, formed by the trace constrained, positive semidefinite cone (i.e., $X \succcurlyeq 0$ & $X^* = X, \operatorname{tr} X \leq 1$), and f is convex differentiable in X. This template clearly can be put to the form of (3.1) by using the so-called Burer-Monteiro splitting $X = UU^*$ [BM03b, BM05]. The template (P) above includes the standard semidefinite programming as a special case, with $g(X) = \operatorname{tr} DX$.

Many studies (e.g., see [Rag08]) have revealed a surprising phenomenon that, for many of these problems, one cannot obtain better approximations beyond the semidefinite relaxation, whose optimal approximation bound is governed by the Grothendieck constant [Gro96].

Semidefinite convex optimization problems often have low-rank solutions that can be represented with $\mathcal{O}(n)$-storage as they are motivated by the template (3.1). However, semidefinite programming methods require us to store a matrix variable with size $\mathcal{O}(n^2)$, which prevents the application of virtually all convex methods at large scale.

Indeed, storage, and not necessarily the computation, is the main obstacle that can prevent us from solving large-scale semidefinite problems, including the ones that are not derived from convex relaxations. A key challenge in optimization is therefore to design algorithms whose working space is within a constant factor of the memory required to store the problem instance and its solution.

As a result, there is a major trend in which the literature focuses on algorithms for (3.1): *cf.*, [BM03a, JNS13, Bou15, BKS16, CLS15]. For the most part, the connection to the semidefinite programming is based on the idea of Burer-Monteiro [BM03b, BM05], which relate the matrix variable $X = UU^*$ and the sets $\mathscr{U}$ and $\mathscr{X}$.

A variety of optimization schemes can be applied to (3.1), including quasi-Newton methods, (stochastic) gradient descent and its variants, and manifold optimization techniques. Some methods also have the auxiliary benefit of local linear convergence under restricted cases. One key idea, already present in [BM03b, BM05], is to set r to the Barvinok-Pataki bound [Pat98, Bar95] to rule out the possibility of local minima. This approach ensures that any stationary point is as good as the global minima. However, we still have the challenge of finding a stationary point.

The main difficulty in solving the template (3.4) is nonlinear constraint along with a non-convex objective function without error bound conditions. This work precisely addresses this challenge with a simple optimization framework relying on augmented Lagrangian (ALM) [Hes69, Pow78, Ber14] and matrix factorization techniques without assuming convexity, linearity of constraints and error bounds on the objective funcion.

We note that as mentioned in [BM05] and common in non-convex optimization literature [BBJN18, JGN$^+$17], some perturbation to the gradient or the augmented Lagrangian function is needed to rule out saddle points and prove convergence to local minima. However, for the specific type of non-convexity introduced by the factorization, as noted in [BM05], this perturbation is only needed for theoretical results and in practice, it does not seem to be necessary. Motivated by this

fact, in this work, we focused on first-order stationarity conditions, however, we note that a more delicate analysis with perturbation might enable us to obtain second order stationarity conditions as well.

In a nutshell, our method consists of alternating one primal and one dual update in the augmented Lagrangian framework with a condition for the dual update. Our method is connected to both inexact and linearized-preconditioned splitting methods. As will be made precise in the sequel, our method can be seen as a practical version of the method suggested in [BM03b, BM05] with stronger guarantees.

3.2 Preliminaries

Notation. For a matrices $X, Y \in \mathbb{R}^{n\times n}$, we use $X \succeq 0$ to denote that X is positive semi-definite. We use $\|\cdot\|_F$ to denote the Frobenius norm and $\|\cdot\|_2$ to denote the spectral norm. We use * to denote the conjugate transpose of a matrix and $\langle X, Y\rangle = \operatorname{tr} X^\top Y$ to denote the matrix inner product. We use $\iota_{\mathscr{U}}(x)$ to denote the indicator function for a set $\mathscr{U}$, $\mathscr{P}_{\mathscr{C}}(X)$ to denote the projection of X to $\mathscr{C}$. We also define the distance of X to $\mathscr{C}$ as $d_{\mathscr{C}}(X) = \inf_{U\in\mathscr{C}} \|U - X\|$. $B(0;\epsilon)$ is the closed ball with radius ϵ. For a convex function g, we denote by $\operatorname{dom}(g)$ its effective domain.

We say that a differentiable function $g : \operatorname{dom} g \to \mathbb{R}$ has L_g-Lipschitz gradient if, $\forall X, Y \in \operatorname{dom} g$, the following holds:

$$\|\nabla g(X) - \nabla g(Y)\| \le L_g \|X - Y\|. \tag{3.2}$$

For an operator $T\colon \mathscr{X} \mapsto \mathscr{Y}$, we sometimes abuse the notation and use TU to mean applying the operator to input U, *i.e.*, $TU = T(U)$.

Lastly, we denote the gradient mapping as

$$\mathscr{G}_{\beta,\gamma}(U, Y) = \gamma^{-1}(U - U^+), \tag{3.3}$$

where U^+ denotes the next iterate.

Assumption 1. *The function g is proper, closed and convex.*

(i) There exist a τ_X-Lipschitz continuous operator $T\colon U \mapsto T(U)$ and a bounded linear operator A such that $TU = AX$.

(ii) Linear operator A^* has a trivial null space.

3.2.1 Factorization of the problem

Given a convex semi-definite problem as (P), to remove the constraint $X \succcurlyeq 0$, a factorization of the decision variable as $X = UU^*$, where $U \in \mathbb{R}^{n\times r},\ r \ll n$, gives the following formulation:

$$\underset{U\in\mathbb{R}^{n\times r}}{\text{minimize}} \left\{ f(U) : \mathscr{A}UU^* = b \quad \text{and} \quad U \in \mathscr{U} \right\} \tag{3.4}$$

There are several well-known advantages of such a non-convex formulation, including its low dimensionality as well as the removal of difficult-to-project nuclear or trace norm constraints.

3.2.2 Augmented Lagrangian

We follow a classical approach to use the augmented Lagrangian formulation [Hes69, Pow78, Ber14] to split the linear constraint in (3.1) and form the saddle point problem.

$$\max_{Y\in\mathbb{R}^m} \min_{U\in\mathscr{U}} \left\{ \mathscr{L}_\beta(U,Y) := f(U) + \langle Y, TU - b\rangle + \frac{\beta}{2}\|TU-b\|^2 \right\},$$

where the term $\frac{\beta}{2}\|TU-b\|^2$ is referred to as the augmented term.

The main intuition behind ALM is to apply gradient ascent to the dual problem $\max_{Y\in\mathbb{R}^m} d(Y)$ where $d(Y) := \min_{U\in\mathscr{U}} \mathscr{L}_\beta(U,Y)$. Note that $\mathscr{L}_\beta(U,Y)$ has Lipschitz continuous gradient in Y with constant β (for a proof, see [HL17]), so the dual ascent step takes the form

$$Y_{k+1} \leftarrow Y_k + \beta_k \nabla d(Y_k),$$

where $\nabla d(Y_k) = (TU^*(Y_k) - b)$ and

$$U^*(Y_k) = \arg\min_{U\in\mathscr{U}} \mathscr{L}_{\beta_k}(U, Y_k). \tag{3.5}$$

We can now see that to be able to get an estimate of the gradient of the dual function, one needs to solve the problem $\min_{U\in\mathscr{C}} \mathscr{L}_{\beta_k}(U, Y_k)$ which is often referred to as the primal update step in ALM. Since solving this problem is not always simple in practice, several inexact and linearized versions of ALM has been proposed in the literature.

There exists methods that are using a constant penalty parameter $\beta_k = \beta$ or adaptive β_k that changes through the iterations. For the constant penalty parameter scheme, tuning the parameter is not always straightforward in practice. Therefore, we focus on adaptive penalty parameter β_k which both removes this drawback and is also an important component in our analysis.

	T: Linear	T: Nonlinear	CS*: Smooth manifold	CS*: Not smooth manifold	Algorithm
[BM05]	✓	✓	✓	✓	Simple
[BMAS14, BAC16]	✓	✓	✓	✗	Complicated
[BGM+16]	✓	✓	✓	✓	Complicated
[WYZ15, LSG17]	✓	✗	✓	✓	Simple
Our method	✓	✓	✓	✓	Simple

Table 3.1 – A comparison with existing work. This comparison is for solving the SDP: $\{\min f(X) : X \in \mathbb{S}^n_+, A(X) = b\}$, where $\mathbb{S}^n_+$ is the set of all symmetric positive semi-definite matrices. We use the splitting $X = UU^*$ to cast this problem into template (3.1). *CS: constraint set refers to the set $\{X : X \in \mathbb{S}^n_+, A(X) = b\}$

3.2.3 Characterization of a stationary point

Following the ALM literature, we call $\{U^\star, Y^\star\}$ as a stationary point if

$$0 \in \nabla \mathscr{L}_\beta(U^\star, Y^\star) + \partial \iota_{\mathscr{U}}(U^\star) = \nabla f(U^\star) + A^* Y^\star U^\star + \partial \iota_{\mathscr{U}}(U^\star) \tag{3.6}$$

$$0 = TU^\star - b \tag{3.7}$$

For finding a stationary point, in our analysis, we will show the convergence of both the gradient mapping and the feasibility to 0. In the sequel, we assume that such a saddle point exists. Note that these conditions are also referred as the Karush-Kuhn-Tucker (KKT) conditions.

Related works. Augmented Lagrangian based methods are first proposed in [Hes69, Pow78]. In the convex setting, ALM is studied extensively [Ber99, Ber14, LM16, NNTD14]. There exists inexact and linearized versions of ALM which aims at solving (3.5) more efficiently. Some works also considered the application of ALM/ADMM to non-convex problems [WYZ15, LSG17]. These works assume that the operator in (3.1) is linear, therefore, they do not apply to our setting since we have a nonlinear constraint in addition to a non-convex objective function.

Series of influential papers from Burer and Monteiro [BM03b, BM05] proposed using the splitting $X = UU^*$ and they also suggested solving the problem using ALM. First, they did not have any inexact analysis. In other words, their analysis requires primal subproblems to be solved exactly which is not practical. Their practical stopping condition is also not analyzed theoretically. Secondly, they have to put an artificial bound to the primal domain which will be ineffective in practice which is impossible to do without knowing the norm of the solution and their results do not extend to the case where projection in primal domain are required in each iteration. Lastly, their results are for convergence, without any rate guarantees.

The authors focused on the special case of SDPs without linear constraints in [BKS16]. They

prove the convergence of gradient descent on Burer-Monteiro factorized formulation. In their followup work [PKB+16], the authors considered projected gradient descent, but only for strongly convex functions. Their results are not able to extend to linear constraints and general convex functions, therefore not applicable to general SDPs in the form of (P).

Another line of work focused on solving a specific kind of SDPs focused on applying gradient descent or trust regions methods on manifolds [BMAS14, BAC16]. The authors show that they can apply gradient descent on manifolds to satisfy the first order stationarity conditions in $\mathcal{O}(1/\epsilon^2)$ iterations. In addition, they apply trust regions methods on manifolds to satisfy the second order stationarity conditions in $\mathcal{O}(1/\epsilon^3)$ iterations. Firstly, these methods have to assume that the problem will be on a smooth manifold, which holds for Maximum Cut and generalized eigenvalue problems, but is not satisfied for other important SDPs such as clustering, quadratic programming (QAP) and optimal power flow. Secondly, as noted in [BAC16], the per iteration cost of their method for Max-Cut problem is $\mathcal{O}(n^6)$ for solving (3.1) which is astronomically larger than our cost of $\mathcal{O}(n^2 r)$ where $r \ll n$.

Another recent line of work [CMV19] focused on solving the nonlinear constrained non-convex problem template (3.4) by adapting the primal-dual method of Chambolle and Pock [CP11]. The authors proved the convergence of the method with rate guarantees by assuming error bound conditions on the objective function. They do not apply to the general semidefinite programming since $f(U) = \langle C, UU^* \rangle$.

[BBJN18] focused on the penalty formulation of (3.1) and studied the optimality of second order stationary points of the formulation. However, their results are for connecting the stationary points of the penalty formulation of (3.1) to the penalty formulation of (P) and not to the constrained problem itself.

The method presented in [BGM+16] can also handle the same problem but their algorithm is much more complicated than ours.

Table 3.1 provides a comparison of our work with the related methods in the literature which we have summarized in this subsection.

3.3 Algorithm & Convergence

In this section, we present our algorithm along with its convergence guarantees. We denote the Lipschitz constant of $\nabla_U \mathcal{L}_\beta(U, Y)$ as L_β.

Algorithm 2

Input: Choose $\beta_{-1}, \beta_1 > 0, \alpha > 1, c_{10} \geq 10\beta_0\|TU_0 - b\|^2$, $\gamma_k = \frac{1}{L_{\beta_k}}$, where L_{β_k} is given in (3.10) and $U_0 \in \mathscr{U}, Y_0 = 0$.

for $k \leftarrow 0, 1, \cdots, k_{\max}$ **do**
  $s = 1$
  while $\left(\frac{\beta_{k-1}}{2} + \frac{\beta_{k,s}}{2}\right)\|TU_k - b\|^2 \geq \frac{2}{4\gamma_k}\|U_{k+1,s} - U_k\|^2 + \frac{c_{k,s}}{(k+1)^\alpha}$ **do**
    $U_{k+1,s} \leftarrow \mathscr{P}_{\mathscr{C}}(U_k - \gamma_k \nabla \mathscr{L}_{\beta_{k,s}}(U_k, Y_k))$
    Update $\beta_{k,s+1} \leq \beta_{k,s}$, $c_{k,s}$, as in Remark 1.
    $s \leftarrow s + 1$
  end while
  Assign $\beta_k = \beta_{k,s}$ and $U_{k+1} = U_{k+1,s}$, $c_{k,s} = c_k$
  $Y_{k+1} \leftarrow Y_k + \beta_k(TU_{k+1} - b)$.
  Update β_{k+1} such that $\beta_{k+1} \geq \beta_k$ as in Remark 1.
end for

Remark 1. *The precise updates for β_k parameter does not affect in our analysis as long as the condition in the while loop holds. In practice, we observed that in the inner loop, decreasing β_k with a small linear factor as $\beta_{k,s+1} = \frac{1}{1.1}\beta_{k,s}$ and in the outher loop, increasing beta in the order of $\Theta\left(\frac{1}{k^{1/3}}\right)$ works well. We start with $\beta_{k,0} > \beta_k$ and $c_{k,s} = c_k$, then decreasing $\beta_{k,s}$ and increasing $c_{k,s}$ with a linear factor as $\beta_{k,s+1} = \frac{1}{\omega_1}\beta_{k,s}$ and $c_{k,s+1} = \omega_2 c_{k,s}$ for some fixed $\omega_1 \in]1, 2[$ and $\omega_2 \in [1, 2[$. We observe that in our experiment, the inner loop terminates after finite number steps even if $\omega_1 = 1$.*

When the while loop condition in our algorithm is satisfied, our algorithm can be viewed as a linearized and preconditioned ALM applied to solve the non-convex and nonlinear problem (3.1). When the condition is not satisfied, our algorithm requires to perform more primal update steps. This behaviour resembles inexact ALM. Thus, our method can be considered as a mix between these two methods.

We now present the convergence analysis of our method. We firstly need the following lemma which shows that $\mathscr{L}_\beta(U, Y)$ has Lipschitz gradient in a bounded domain.

Lemma 3.3.1. *Suppose $\mathscr{L}_\beta(U, Y)$ is differentiable function in U with gradient:*

$$\begin{aligned}\nabla \mathscr{L}_\beta(U, Y) &= \nabla f(U) + DT(U)^* Y + \beta DT(U)^*(TU - b) \\ &= \nabla f(U) + A^* Y U + \beta A^*(A(UU^\top) - b)U,\end{aligned} \tag{3.8}$$

where $DT(U)$ is the jacobian of the nonlinear operator T at U and $A(\cdot)$ is a linear operator such that $A(UU^\top) := TU$. Suppose that $\mathscr{U}$ is bounded, $\forall U \in \mathscr{U}, \|U\|_F \leq \tau$, then, for all U, V in $\mathscr{U}$, we have

$$\|\nabla \mathscr{L}_\beta(U, Y) - \nabla \mathscr{L}_\beta(V, Y)\|_F \leq (L_f + L_\beta)\|U - V\|_F, \tag{3.9}$$

where

$$L_\beta = 6\tau^2\beta\|A\|^2 + 2\|A^*(Y+\beta b)\|. \tag{3.10}$$

Proof. Let us first define

$$g_\beta(U,Y) = \frac{\beta}{2}\|TU-b\|^2 + \langle Y, TU-b\rangle; \text{and that } \mathscr{L}_\beta(U,Y) = g_\beta(U,Y) + f(U). \tag{3.11}$$

For each $V \in \mathscr{U}$, set $W = VV^*$. We have

$$\begin{aligned}
\frac{1}{2}\|\nabla g_\beta(U,Y) - \nabla g_\beta(W,Y)\|_F &= 2\|\nabla g_\beta(X,Y)U - \nabla g_\beta(W,Y)V\|_F \\
&= \|\nabla g_\beta(X,Y)U - \nabla g_\beta(X,Y)V + \nabla g_\beta(X,Y)V - \nabla g_\beta(W,Y)V\|_F \\
&\le \|\nabla g_\beta(X,Y)U - \nabla F_\beta(X,Y)V\|_F + \|\nabla F_\beta(X,Y)V - \nabla F_\beta(W,Y)V\|_F \\
&\le \|\nabla g_\beta(X,Y)\|_2\|U-V\|_F + \|\nabla g_\beta(X,Y) - \nabla g_\beta(W,Y)\|_2\|V\|_F \\
&\le \|\nabla g_\beta(X,Y)\|_2\|U-V\|_F + \tau\|\nabla g_\beta(X,Y) - \nabla g_\beta(W,Y)\|_F.
\end{aligned} \tag{3.12}$$

Since $\nabla\mathscr{L}_\beta$ is $\beta\|A\|^2$-Lipschitz continuous in X, we have

$$\begin{aligned}
\tau\|\nabla g_\beta(X,Y) - \nabla g_\beta(W;Y)\|_F &\le \tau\beta\|A\|^2\|X-Y\|_F \\
&= \tau\beta\|A\|^2\|UU^* - UV^* + UV^* - VV^*\|_F \\
&\le 2\beta\tau^2\|A\|^2\|U-V\|_F.
\end{aligned} \tag{3.13}$$

By the same manner, we also have

$$\begin{aligned}
\|\nabla g_\beta(X,Y)\|_2 \le \|\nabla g_\beta(X,Y)\|_F &\le \|\nabla g_\beta(X,Y) - \nabla g_\beta(0,Y)\|_F + \|\nabla g_\beta(0,Y)\|_F \\
&= \|\nabla g_\beta(X,Y) - \nabla g_\beta(0,Y)\|_F + \|\nabla g_\beta(0,Y)\|_F \\
&\le \beta\tau^2\|A\|^2 + \|\nabla g_\beta(0,Y)\|_F.
\end{aligned} \tag{3.14}$$

Inserging (3.13) and (3.14) into (3.12), we get the result. □

We next recall the relation between the gradient mapping and function value $\mathscr{L}_\beta$ in the following lemma which relies on Lemma 3.3.1.

Lemma 3.3.2. *Let $U \in \mathscr{U}$. Suppose that $\nabla\mathscr{L}_\beta(\cdot,Y)$ is $L_f + L_\beta$-Lipschitz continuous, for any $\gamma \in]0,(1-0.5)/(L_\beta+L_f)[$, we also have*

$$\|\mathscr{G}_{\beta,\gamma}(U,Y)\|_F^2 \le \frac{4}{3\gamma}(\mathscr{L}_\beta(U,Y) - \mathscr{L}_\beta(U^+,Y)) \le \frac{4}{3\gamma}(\mathscr{L}_\beta(U,Y) - \mathscr{L}_\beta(U^\star,Y)), \tag{3.15}$$

where $U^+ = P_{\mathscr{U}}(U - \gamma\nabla\mathscr{L}_\beta(U,Y))$.

Proof. We refer the reader for the proof of Lemma 3.3.2 to [GLZ16] □

Lemma 3.3.3. *Suppose that*

$$\mathscr{G}_{\beta_k,\gamma_k}(U_k, Y_k) \le \epsilon \qquad \text{and} \qquad \|TU_{k+1} - b\| \le \epsilon.$$

Then (U_{k+1}, Y_{k+1}) *is* 2ϵ*-saddle point, i.e* $\|TU_{k+1} - b\| \le 2\epsilon$ *and* $-A^* Y_{k+1} U_{k+1} \in \nabla f(U_{k+1}) + \partial \iota_{\mathscr{U}}(U_{k+1}) + B(0, 2\epsilon)$.

Proof. By the definition of $P_{\mathscr{U}}$, we have

$$(U_k - U_{k+1})/\gamma_k - \nabla \mathscr{L}_{\beta_k}(U_k, Y_k) \in \partial \iota_{\mathscr{U}}(U_{k+1}). \tag{3.16}$$

Adding $\nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k)$ to both sides, we obtain

$$(U_k - U_{k+1})/\gamma_k - \nabla \mathscr{L}_{\beta_k}(U_k, Y_k) + \nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k) \in \partial \iota_{\mathscr{U}}(U_{k+1}) + \nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k). \tag{3.17}$$

Using the Lipschitz continuous of $\nabla \mathscr{L}_{\beta_k}(\cdot, Y_k)$, we get

$$\|\nabla \mathscr{L}_{\beta_k}(U_k, Y_k) - \nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k)\| \le (L_{\beta_k} + L_f)\|U_{k+1} - U_k\| \le \|U_{k+1} - U_k\|/\gamma_k \le \epsilon, \tag{3.18}$$

which implies that

$$\|(U_k - U_{k+1})/\gamma_k - \nabla \mathscr{L}_{\beta_k}(U_k, Y_k) + \nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k)\| \le 2\epsilon. \tag{3.19}$$

Moreover,

$$\begin{aligned} \nabla \mathscr{L}_{\beta_k}(U_{k+1}, Y_k) &= \nabla f(U_{k+1}) + A^*(Y_k + \beta_k(TU_{k+1} - b))U_{k+1} \\ &= \nabla f(U_{k+1}) + A^* Y_{k+1} U_{k+1}. \end{aligned} \tag{3.20}$$

Therefore, we derive from (3.17) that

$$-A^* Y_{k+1} U_{k+1} \in \nabla f(U_{k+1}) + \partial \iota_{\mathscr{U}}(U_{k+1}) + B(0, 2\epsilon). \tag{3.21}$$

This together with $\|TU_{k+1} - b\| \le \epsilon$, imply that (U_{k+1}, Y_{k+1}) is 2ϵ-saddle point. □

Now, we present our main theorem which characterizes the convergence analysis of Algorithm 2.

Theorem 3.3.4. *Assumption 1 is satisfied and* $|\langle \dot{Y}_k \mid AX_k - b \rangle| = \mathscr{O}(\beta_k^s)$ *for some* $s \ge 0$*, and* $(k\beta_k^{-s}\gamma_{k,\min}) \to \infty$*, where* $\gamma_{K,\min} = \min_{0 \le i \le K} \gamma_i$*. In addition,* $(\beta_k)_{k \in \mathbb{N}}$ *is bounded. Then*

$$\min_{0 \le k \le K} \|\mathscr{G}_{\beta_k,\gamma_k}(U_k, Y_k)\|^2 = \mathscr{O}\left(\frac{1}{K\beta_K^{-s}\gamma_{K,\min}}\right) \to 0. \tag{3.22}$$

and

$$\min_{0 \le k \le K} \|AX_k - b\|^2 \le \mathscr{O}\left(\frac{1}{K\beta_K^{-s}\gamma_{K,\min}}\right) + \frac{c}{\beta_k(1+K)^\alpha} \to 0. \tag{3.23}$$

Proof. Based on the inner loop termination rule of our algorithm:

$$\left(\frac{\beta_{k-1}}{2}-\frac{\beta_k}{2}\right)\|TU-b\|^2 \leq \frac{1}{2\gamma_k}\|U_{k+1}-U_k\|^2+\frac{c}{(k+1)^\alpha} \tag{3.24}$$

For any k, we set $A_k = TU_k - b = AX_k - b$ where $X_k = U_k U_K^*$. For any k, it follows from Lemma 3.3.2 that

$$\begin{aligned}(3\gamma_k/4)\|\mathcal{G}_{\beta_k,\gamma_k}(U_k,Y_k)\|^2 &\leq \left(\mathscr{L}_\beta(U_k,Y_k)-\mathscr{L}_{\beta_k}(U_{k+1},Y_k)\right)\\ &= f(U_k)-f(U_{k+1})+\frac{\beta_k}{2}\|A_k\|^2-\frac{\beta_k}{2}\|A_{k+1}\|^2+\langle Y_k, A_k-A_{k+1}\rangle.\end{aligned} \tag{3.25}$$

Rearranging and using Assumption 1, we obtain

$$\begin{aligned}\frac{1}{2}\|A_{k+1}\|^2 &\leq \frac{1}{2}\|A_k\|^2-(3\gamma_k/4\beta_k)\|\mathcal{G}_{\beta_k,\gamma_k}(U_k,Y_k)\|^2\\ &\quad+\frac{1}{\beta_k}(f(U_k))-f(U_{k+1}))+\frac{1}{\beta_k}\langle Y_k, TU_k-TU_{k+1}\rangle\\ &= \frac{1}{2}\|A_k\|^2-(3\gamma_k/4\beta_k)\|\mathcal{G}_{\beta_k,\gamma_k}(U_k,Y_k)\|^2\\ &\quad+\frac{1}{\beta_k}(f(U_k))-f(U_{k+1}))+\frac{1}{\beta_k}\langle Y_k, A_k-A_{k+1}\rangle.\end{aligned} \tag{3.26}$$

Focusing on the last term, we get

$$\begin{aligned}\frac{1}{\beta_k}\langle Y_k, A_k-A_{k+1}\rangle-\|A_k-A_{k+1}\|^2 &= \frac{1}{\beta_k}\langle Y_k-\beta_k(A_k-A_{k+1}), A_k-A_{k+1}\rangle\\ &= \frac{1}{\beta_k}\langle \nabla g_{\beta_k}(X_{k+1},Y_k), X_k-X_{k+1}\rangle-\langle A_k, A_k-A_{k+1}\rangle\\ &= \frac{1}{\beta_k}\langle \nabla g_{\beta_k}(X_{k+1},Y_k), X_k-X_{k+1}\rangle-\|A_k\|^2+\langle A_k, A_{k+1}\rangle,\end{aligned} \tag{3.27}$$

which, by using the elementary inequality $\langle a \mid b\rangle = \frac{1}{2}\|a\|^2+\frac{1}{2}\|b\|^2-\frac{1}{2}\|a-b\|^2$, implies that

$$\begin{aligned}&\frac{1}{\beta_k}\langle Y_k, A_k-A_{k+1}\rangle-\frac{1}{2}\|A_{k+1}\|^2+\frac{1}{2}\|A_k\|^2\\ &\quad\leq \frac{1}{\beta_k}\langle \nabla g_{\beta_k}(X_{k+1},Y_k), X_k-X_{k+1}\rangle+\frac{1}{2}\|A_{k+1}-A_k\|^2.\end{aligned} \tag{3.28}$$

It follows from the convexity of g_{β_k} that

$$\begin{aligned}&\frac{1}{\beta_k}\langle \nabla g_{\beta_k}(X_{k+1},Y_k), X_k-X_{k+1}\rangle+\frac{1}{2}\|A_k-A_{k+1}\|^2\\ &\qquad\leq \frac{1}{\beta_k}g_{\beta_k}(X_k,Y_k)-\frac{1}{\beta_k}g_{\beta_k}(X_{k+1},Y_k)\end{aligned} \tag{3.29}$$

Now we change the dual variable:

$$\begin{aligned}\frac{1}{\beta_k} g_{\beta_k}(X_k, Y_k) - \frac{1}{\beta_k} g_{\beta_k}(X_k, Y_{k-1}) &= \frac{1}{\beta_k}\langle Y_k - Y_{k-1}, AX_k - b\rangle \\ &\leq (\beta_{k-1}/\beta_k)\langle AX_k - b \mid AX_k - b\rangle \\ &= (\beta_{k-1}/\beta_k)\|AX_k - b\|^2, \end{aligned} \tag{3.30}$$

We combine (3.25), (3.29) and (3.30) to get,

$$\begin{aligned} g_{\beta_k}(X_{k+1}, Y_k) &\leq g_{\beta_k}(X_k, Y_{k-1}) + (f(U_k)) - f(U_{k+1})) \\ &\quad + \beta_{k-1}\|AX_k - b\|^2 - (3\gamma_k/4)\|G_{\beta_k,\gamma_k}(U_k; \dot{Y}_k)\|^2 \\ &= g_{\beta_{k-1}}(X_k, Y_{k-1}) + (f(U_k)) - f(U_{k+1})) \\ &\quad + (\frac{\beta_{k-1}}{2} + \frac{\beta_k}{2})\|AX_k - b\|^2 - (3\gamma_k/4)\|\mathscr{G}_{\beta_k,\gamma_k}(U_k; \dot{Y}_k)\|^2 \\ &\leq g_{\beta_{k-1}}(X_k, Y_{k-1}) + (f(U_k)) - f(U_{k+1})) + \frac{c}{(k+1)^\alpha} - (\gamma_k/4)\|\mathscr{G}_{\beta_k,\gamma_k}(U_k, Y_k)\|^2. \end{aligned} \tag{3.31}$$

which implies that

$$K \min_{1\leq i\leq K-1} \gamma_i \|\mathscr{G}_{\beta_i,\gamma_i}(U_i, Y_i)\|^2 \leq \mathscr{L}_{\beta_1}(X_1, Y_0) - \mathscr{L}_{\beta_K}(X_K, Y_{K-1}) + \sum_{i=1}^{K-1} \frac{c}{(i+1)^\alpha} \tag{3.32}$$

$$\leq S - \langle Y_K, AX_K - b\rangle, \tag{3.33}$$

where $S = \mathscr{L}_{\beta_1}(X_1, Y_0) - f(U_k) + \sum_{i=1}^{K-1} \frac{c}{(i+1)^\alpha}$.

Hence,

$$\min_{0\leq k\leq K} \|\mathscr{G}_{\beta_k,\gamma_k}(U_k, Y_k)\|^2 \leq \frac{S - \langle Y_K, AX_K - b\rangle}{K} = \mathscr{O}\left(\frac{1}{K\beta_K^{-s}\gamma_{K,min}}\right) \tag{3.34}$$

The second conclusion follows from (3.24). □

Remark 2. *The choice of s determines the strictness of our assumptions. Choosing $s = 0$ gives the best rate in theory, however it requires boundedness of the dual domain. Choosing $s = 1$ gives a growth condition in the dual variable by guaranteeing $\mathscr{O}\left(\frac{1}{k^{1/3}}\right)$ rate for both gradient mapping and feasibility by choosing $\beta_k = \Theta\left(\frac{1}{k^{1/3}}\right)$. In the presentation of our algorithm and the numerical experiments, we use $s = 1$ everywhere without loss of generality.*

As we will also observe in the experimental results, dual update step is crucial for a practically

fast algorithm. Our theory also supports the importance of that dual variable updates.

Extension to general constraints. For clarity, we presented our algorithm and guarantees for the linear equality constraints. Note however, our algorithm and convergence guarantees can be extended to general constraints, including bounded convex sets or convex cones to solve

$$\underset{U\in\mathbb{R}^{n\times r}}{\text{minimize}}\left\{f(U) : TU - b \in \mathcal{K} \quad \text{and} \quad U \in \mathcal{U}\right\} \tag{3.35}$$

where $\mathcal{K}$ is a convex set. Our results can be extended to general constraints in a straightforward manner using a similar approach as in Section 5.3 of [TDFC18].

3.4 Numerical experiments

In this section, we consider the application of our algorithm to semidefinite programming. We consider generalized eigenvalue and clustering problems. For the former, we compare our algorithm to ManOpt [BMAS14, BAC16] and Burer-Monteiro's algorithm [BM03b, BM05]. ManOpt does not apply to the latter. Hence, we compare against Burer-Monteiro(abbreviated as BM) and a recently proposed storage efficient convex method HCGM [YFLC18].

An important note here is that our algorithm works with $U \in \mathbb{R}^{n\times r}$ variable which has reduced dimension and we only apply matrix matrix multiplications with at most $\mathcal{O}(n^2 r)$ cost. In contrast, ManOpt involves solving linear systems at each iteration which is handled by Matlab's backslash operator for solving generalized eigenvalue problem. For numerical experiments, we used ManOpt's open source software package for Matlab. BM and our algorithm are also implemented on MATLAB. We refer the reader to [BM03b] for the BM algorithm. Subproblems in the BM-algorithm are terminated as indicated in [BM05].

3.4.1 Maximum Cut

Given a graph, a cut is any partition of the nodes to two disjoint subsets. The size or weight of the cut is defined by the edges that are connecting the nodes in two sets. Max-Cut is an NP-hard problem where the aim is to find the cut with the maximum size or weight. Convex semidefinite relaxation of this problem can be stated as:

$$\min_{X\in\mathbb{S}^{n\times n}} \langle C, X\rangle \quad \text{s.t.} \quad \operatorname{diag}(X) = 1, \operatorname{tr} X = n, X \succeq 0 \tag{3.36}$$

Introducing non-convex splitting $X = UU^*$ we use the following relaxed problem,

$$\min_{U\in\mathbb{R}^{n\times r}} \langle C, UU^*\rangle \quad \text{s.t.} \quad \operatorname{diag}(UU^*) = 1, \|U\|_F \le \sqrt{n}. \tag{3.37}$$

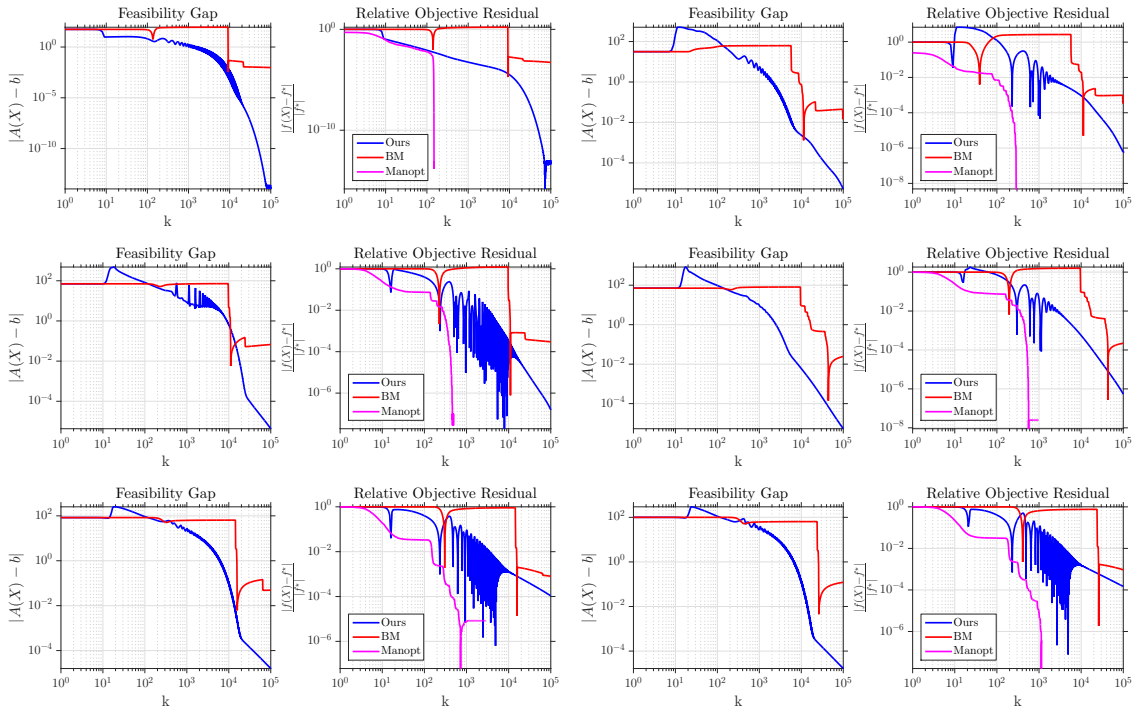

Figure 3.1 – Performance of 3 algorithms for solving Max-Cut on 6 datasets. From top-left to bottom-right: G50, G54, G56, G59, G62, G67

For this test, we use benchmark sparse graph data from [DH11] which is used in the literature [BVB16]. The dimensions of the graphs we used are given in the following table:

Table 3.2 – Datasets used for Max-Cut.

Datasets	G50	G54	G56	G59	G62	G67
n	3,000	1,000	5,000	5,000	7,000	10,000

In this experiment, we compare Burer-Monteiro's method, ManOpt and our method. We use the implementation of ManOpt from [BMAS14] and implemented Burer-Monteiro and our method in MATLAB. Among these, ManOpt has a per iteration cost of $\mathcal{O}(n^6)$ [BAC16] whereas the other two methods has the cost of $\mathcal{O}(n^2)$ As can be seen from the plots, we consistently outperform Burer-Monteiro's method both in terms of feasibility and objective residual in terms of the iterations. Even though ManOpt has faster convergence in terms of iterations, because of its dimension dependence, our method is faster in terms of overall time cost.

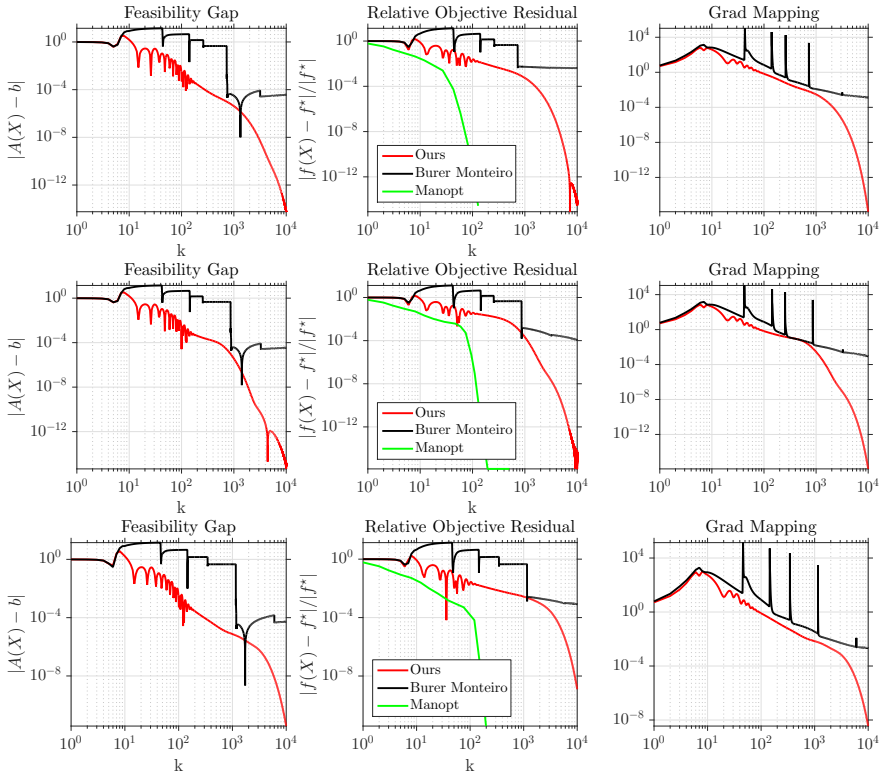

Figure 3.2 – Performance of 3 algorithms for solving generalized eigenvalue problem. Problem sizes from top to bottom: $n = 6000$, $n = 7000$, $n = 10000$

3.4.2 Generalized eigenvalue problem

Generalized eigenvalue problem has extensive applications in machine learning, statistics and data analysis [GJN+16]. The well-known non-convex formulation of the problem is [BVB16].

$$\min_{x\in\mathbb{R}^n} x^\top Cx \quad \text{s.t.} \quad x^\top Bx = 1, \tag{3.38}$$

where B, C are symmetric matrices. Semidefinite relaxation of this problem gives

$$\min_{X\in\mathbb{S}^{n\times n}} \langle C, X\rangle \quad \text{s.t.} \quad \langle B, X\rangle = 1, X \succeq 0. \tag{3.39}$$

Another common assumption is $B \succ 0$. Due to the invertibiilty of B, we have $\langle U, BU\rangle \geq \frac{\|U\|_F^2}{\|B^{-1}\|}$, which implies the constraint $\|U\|_F^2 \leq \|B^{-1}\|$. Using Burer-Monteiro splitting with this fact gives the problem

$$\min_{U\in\mathbb{R}^{n\times r}} \langle U, CU\rangle \quad \text{s.t.} \quad \langle U, BU\rangle = 1, \|U\|_F \leq \|B^{-1}\|^{(1/2)}. \tag{3.40}$$

We randomly generated the matrices C and B. We then set $||B||_F = 1$. For different problem

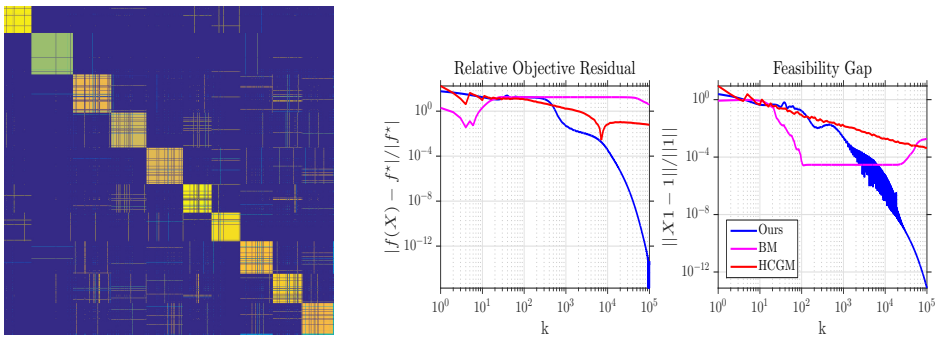

Figure 3.3 – Left: Clustering result of our algorithm. Right: Performance of 3 algorithms for solving clustering on MNIST dataset.

sizes, we tuned the β_1 in Algortihm 2 and initial penalty parameter for BM's algorithm.We used ManOpt's open source software package. We refer the reader to [BM03b, BM05] for the Burer-Monteiro algorithm. As can be seen from Figure 3.2, our algorithm consistently outperforms BM. Also note that ManOpt involves solving linear systems in B at each iteration.

3.4.3 Clustering

We would like to solve the following semidefinite relaxation of model-free k-means clustering:

$$\min_{X \in \mathbb{S}^{n \times n}} \langle C, X \rangle \quad \text{s.t.} \quad X1 = 1, X \geq 0, \operatorname{tr} X = \rho, X \geq 0 \tag{3.41}$$

where C is the ℓ_2-distance metric. For non-convex factorization of this problem, we have a few choices. Firstly, one can join the linear constraints $X1 = 1$ and $X \geq 0$. However, in convex relaxation, $X \geq 0$ constraint is required because of the nonnegativity of factors U. Therefore, we choose to require $U \geq 0$ even though it does not map directly to the convex problem, it seems to be tighter. Next, $\operatorname{tr} X = k$ constraint is translated to $\|U\|_F = \sqrt{k}$. However, this constraint is not convex, therefore we cannot directly project to it. We can either join these constraints to nonlinear constraints or use a relaxation $\|U\|_F \leq \sqrt{k}$. In the formulation below, we decided to use the latter.

$$\min_{U \in \mathbb{R}^{n \times r}} \langle C, UU^* \rangle \quad \text{s.t.} \quad (UU^*)1 = 1, U \geq 0, \|U\|_F \leq \rho, \tag{3.42}$$

For this problem, we have used a synthetic dataset in addition to the classical MNIST dataset.

For the test with MNIST data, similar to [YFLC18], we use the test setup and the preprocessed data by [MVW17]. For the preprocessing details, please see [MVW17]. 1000 data points are used for the test which means $n = 1000$ in our template. Elements of matrix C is composed of the pairwise Euclidean distances of the data points.

3.5 Bibliographic notes

The initial version of this chapter appeared in Workshop on Modern Trends in Non-convex Optimization for Machine Learning, ICML, 2018. It has been revised by the author of this dissertation to show that approximate stationary points in terms of gradient mapping implies approximate saddle points under mild assumptions. Theorem 3.3.4 is in part due to Bang Cong Vu.

4 Linearized ADMM for Non-convex Problems with Nonlinear constraints

This chapter studies an increasingly important problem of minimizing two sets of variables with the separable non-convex composite objective with nonlinear constraints.

We introduce a linearized alternating direction method of multipliers (ADMM) framework that only requires two consecutive proximal gradient steps on the primal variables and a gradient ascent step in the dual. The proposed scheme achieves $\epsilon-$first order stationarity by reducing the feasibility and gradient mapping at a rate $\tilde{\mathcal{O}}(\frac{1}{\epsilon^4})$ subject to a regularity condition on the constraints. We also establish the same complexity result to ϵ approximate KKT points of the constrained problem. Our analysis allows us to recover the known convergence rates with a single loop algorithm, which is simpler than the inexact Augmented Lagrangian variants.

Our framework also handles the linearized augmented Lagrangian as a special case. Numerical evidence on large-scale non-convex machine learning problems (such as continuous relaxation of causal learning problem, SDP relaxation of clustering and Max-Cut problems) show that the algorithm is scalable and accurate while requiring very little tuning.

4.1 Introduction

Many important problems in machine learning [MNS15, SSGB07], signal processing, communication [Sin11, SS11, LTHC20] and theoretical computer science [KN12, Lov03, ZKRW98] can be captured by the following non-convex optimization template

$$\min_{x\in\mathbb{R}^d} f(x)+g(x) \qquad \text{s.t.} \qquad A(x)=0, \tag{4.1}$$

where $f:\mathbb{R}^d\to\mathbb{R}$ is a twice differentiable function whose gradient is given by $\nabla f(x)\in\mathbb{R}^d$; $g:\mathbb{R}^d\to\mathbb{R}\cup\{\infty\}$ is a proximal-friendly, (possibly) non-differentiable, proper, closed and convex function; $A:\mathbb{R}^d\to\mathbb{R}^m$ is a twice-differentiable mapping whose Jacobian is denoted as $\mathbf{D}A(x)\in\mathbb{R}^{m\times d}$. We additionally assume f and A satisfies the smoothness property such that

$$\|\nabla f(x)-\nabla f(x')\|\le\lambda_f\|x-x'\|, \quad \|\mathbf{D}A(x)-\mathbf{D}A(x')\|\le\lambda_A\|x-x'\|, \tag{4.2}$$

for $\lambda_f, \lambda_A \geq 0$ and every $x, x' \in \mathbb{R}^d$.

The augmented Lagrangian method [Hes69] provides a powerful framework to solve (4.1). Indeed, for $\beta > 0$, the first order stationarity conditions of the constrained problem (4.1) coincides with

$$\min_x \max_y \mathscr{L}_\beta(x, y) + g(x) := f(x) + \langle A(x), y\rangle + \frac{\beta}{2}\|A(x)\|^2 + g(x), \tag{4.3}$$

where $y \in \mathbb{R}^m$ is the dual variable and $\mathscr{L}_\beta(x, y)$ is the augmented Lagrangian corresponding to (4.1). The equivalent reformulation in (4.3) naturally suggests the following iterative algorithm

$$x_{k+1} \in \operatorname*{argmin}_x \mathscr{L}_\beta(x, y_k) + g(x), \qquad y_{k+1} = y_k + \sigma_k A(x_{k+1}), \tag{4.4}$$

to solve (4.1), where σ_k is the dual step size parameter. Every primal update requires to (exactly) solve an unconstrained minimization problem to first (or second) order stationarity. Even in the convex case, calculating the exact solution of the primal subproblem in (4.4) is not practical. Hence, there is a line of work to develop inexact version of this conceptual algorithm both in the convex [NNTD14, TDFC18] and non-convex [SAL$^+$19, LCL$^+$21] optimization setting.

When f is convex and A is linear, certain linearization techniques can be used to approximate the augmented Lagrangian function in the subproblem in 4.4. This allows to obtain a closed form solution to primal minimization problem making the algorithm fully implementable while preserving the global convergence guarantees [YY13]. [LSG19] used similar approaches to obtain a linearized augmented Lagrangian method for a non-convex f and linear A. To our knowledge, this is the first work which studies the hard problem of convergence and complexity of a *linearized* AL method for non-convex optimization with nonlinear constraints.

The classical alternating direction method of multipliers (ADMM) has received significant attention due to its success in solving large scale problems by splitting them into series of subproblems, each of which is much easier to solve. The classical ADMM has been extensively studied for optimization problems with convex constraints [LSG19, BN18, HLR16]. However, many important problems require to handle nonlinear constraints, such as k-means clustering [SAL$^+$19], causal learning [ZARX18] and robustness verification of neural networks [BIL$^+$16, LAL$^+$21]. In this paper, we extend the classical ADMM to handle nonlinear constraints where there are two blocks of variables which are nonlinearly coupled to each other.

We summarize our contributions in the sequel: (i) As the basis of our analysis, we first introduce the Linearized Augmented Lagrangian algorithm (LAL), which, to our knowledge, is the first *single loop* algorithm for this template subject to a regularity condition. (ii) We then prove that LAL achieves the first-order stationarity for (4.1) at the rate of $1/k^{\frac{1}{4}}$. The resulting algorithm is simple to implement and has only one tuning parameter, making it practical for large scale applications.(iii) Then, we extend the same convergence guarantees for the alternating direction method-of-multipliers (ADMM) variant, which is better suited for a variety of problems that require variable splitting. (iv) The success of LAL relies on a geometric regularity condition,

detailed in Section 4.2, which is closely related to Polyak-Lojasiewicz [KNS16], Mangasarian-Fromovitz [Ber14], and other existing conditions in the literature of non-convex programming [BST18, XY17, Roc93, FBFBV12]. We also verify this regularity condition in several important examples.

Roadmap. We first give the necessary notations and preliminaries in Section 4.1.1. Section 4.2 introduces the geometric regularity condition which is vital for the success of our algorithm. Section 4.3 studies the linearized Augmented Lagrangian algorithm along with the convergence rates. In section 4.4, we introduce the linearized ADMM algorithm and its convergence results. Section 4.5 is devoted for literature review.

4.1.1 Preliminaries

Notation. For a cone C, we denote its polar by C^*, namely, $C^* = \{x : \langle x, x' \rangle \leq 0,\ \forall x' \in C\}$. For a set C', its conic hull is defined as $\text{cone}(C') := \bigcup_{\alpha \geq 0} \{\alpha x : x \in C'\}$.

Optimality conditions. Necessary optimality conditions for (4.1) are well-understood. Indeed, x is a first-order stationary point of (4.1) if there exists $y \in \mathbb{R}^m$ for which

$$-\nabla f(x) - \mathbf{D}A(x)^\top y \in \partial g(x), \text{ and } \quad A(x) = 0. \tag{4.5}$$

Recalling (4.3) and for any $\beta > 0$, we observe that (4.5) is equivalent to

$$-\nabla_x \mathscr{L}_\beta(x, y) \in \partial g(x), \text{ and } \quad A(x) = 0, \tag{4.6}$$

which is in turn the first-order optimality condition for (4.3).

Definition 2 (ϵ-KKT points)**.** *$x \in \mathbb{R}^d$ is an ϵ-KKT point if there exists a vector $y \in \mathbb{R}^m$ such that*

$$dist\big(\mathbf{0}, \nabla f(x) + \mathbf{D}A(x)^\top y + \partial g(x)\big) \leq \epsilon, \quad \|A(x)\| \leq \epsilon \tag{4.7}$$

Feasible set of the problem is given as $\mathscr{F} := \{x \in \mathbb{R}^d :\ A(x) = 0\}$. Inspired by [BST18], Definition 2, we define the slightly modified variant of the information zone with additional norm constraint;

Definition 3. ***(Bounded information zone)*** *Given the feasible set $\mathscr{F}$, the bounded information zone for the model problem* (4.1) *is*

$$\mathscr{F} \subset \{x : \|A(x)\| \leq \rho, \|x\| \leq \rho'\} := X_{\rho,\rho'} \subset \mathbb{R}^d \tag{4.8}$$

Note that $X_{\rho,\rho'}$ is the enlargement of the feasible set and a subset of $\mathbb{R}^d$. This is the region where the assumptions we made hold. [BST18] proposes a finite time algorithm to reach such a region.

Technical lemmas. The following technical result shows that the augmented Lagrangian, defined in (4.3), is strongly smooth, see Section 4.12.

Lemma 4.1.1 (Smoothness)**.** *Assume $\mathscr{L}_\beta(.,y)$ is twice continuously differentiable. For all $x, x' \in X_{\rho,\rho'}$ and fixed $y \in \mathbb{R}^m$ and $\beta, \rho, \rho' \geq 0$, it holds that*

$$\|\nabla_x \mathscr{L}_\beta(x,y) - \nabla_x \mathscr{L}_\beta(x',y)\| \leq \lambda_\beta \|x - x'\|, \tag{4.9}$$

where λ_f, λ_A were defined in (4.2), and

$$\lambda_\beta := \lambda_f + \sqrt{m}\lambda_A\big(\|y\| + \beta\rho\big) + \beta d\lambda_A'^2, \qquad \lambda_A' := \max_{\|x\| \leq \rho'} \|\mathbf{D}A(x)\|. \tag{4.10}$$

For a sufficiently small step size γ, the gradient mapping controls the descent in the objective function of (4.3). The following standard result from [BST14] (see remark $4(i)$) formalizes this notion. Section 4.13 contains the proof of the lemma for completeness.

Lemma 4.1.2 (Sufficient decrease property)**.** *For $x \in \mathbb{R}^d$ and $y \in \mathbb{R}^m$, let $x^+ = P_{\gamma,g}(x - \gamma\nabla_x \mathscr{L}_\beta(x,y))$, where $\gamma < 1/\lambda_\beta$, see (4.9). For $x, x^+ \in X_{\rho,\rho'}$ and $\rho, \rho' \geq 0$, it holds that*

$$\|G_{\beta,\gamma}(x,y)\|^2 \leq \frac{2}{\gamma}(\mathscr{L}_\beta(x,y) + g(x) - \mathscr{L}_\beta(x^+,y) - g(x^+)). \tag{4.11}$$

In practice, determining λ_β (and thus the step size γ) by computing the right-hand side of (4.10) is often intractable, since $\lambda_f, \lambda_A, \lambda_A'$ are usually unknown. Instead, we can resort to the line search technique, reviewed below and proved in Section 4.14.

Lemma 4.1.3 (Line search)**.** *Fix $\theta \in (0,1)$ and $\gamma_0 > 0$. For $\gamma' > 0$, let $x_{\gamma'}^+ = P_{\gamma',g}(x - \gamma'\nabla_x \mathscr{L}_\beta(x,y))$, and define*

$$\gamma := \max\Big\{\gamma' = \gamma_0\theta^i : \mathscr{L}_\beta(x_{\gamma'}^+, y) \leq \mathscr{L}_\beta(x,y) + \Big\langle x_{\gamma'}^+ - x, \nabla_x \mathscr{L}_\beta(x,y)\Big\rangle + \frac{1}{2\gamma'}\|x_{\gamma'}^+ - x\|^2\Big\}. \tag{4.12}$$

Then, (4.11) holds and we moreover have that $\gamma \geq \frac{\theta}{\lambda_\beta}$.

4.2 Geometric Regularity

Regularity conditions play a key role in the primal dual analysis of nonlinear optimization problems. Indeed, they are necessary to draw the connection between the KKT conditions and the optimal points of the constrained problem [MF67]. Examples include linear independence constraint qualification (LICQ) and Mangasarian-Fromovitz constraint qualification (MFCQ) [Ber14]. In particular, LICQ implies that gradient vector of each constraint is linearly independent at the primal optimal point $x^\star$. Even in the convex case, KKT conditions are not sufficient for optimality. In other words, a KKT point is not necessarily optimal for the constrained optimization. For example, Slater's condition is sufficient condition for strong duality [BBVP04] in the convex

case which ensures optimality of the KKT points.

Similarly, to successfully solve problem (4.1) in the presence of nonlinear constraints, we require the following condition which is inspired by the Slater's condition and extended to the non-convex setting.

Definition 4. ***(Geometric regularity)*** *In problem (4.1) with $m \leq d$, for a subspace $S \subset \mathbb{R}^d$, let*

$$\nu \|A(x)\| \leq \left\| P_S P_{\text{cone}(\partial g(x))^*} (\mathbf{D}A(x)^\top \cdot A(x)) \right\| \tag{4.13}$$

where $\nu \in \mathbb{R}^+$, $\text{cone}(\partial g(x)$ is the conic hull of the subdifferential $\partial g(x)$ and $P_{\text{cone}(\partial g(x))^*}$ projects onto the polar of this cone, see section 4.1.1. Moreover, $\mathbf{D}A(x)$ is the Jacobian of A. We say that (4.1) satisfies the geometric regularity if $\nu > 0$.

A similar regularity condition is assumed in [SAL+19] and [LCL+21] and it can be obtained as an application of Definition 4. We believe 4.13 is a more compact and more general version of that condition which allows us to draw certain connection to the Slater's condition as well as other regularity conditions in the non-convex programming (see Section 4.8).

Subspace S**.**Role of the subspace S in (4.13) is also to broaden the applicability of the geometric regularity. In particular, when $S = \mathbb{R}^d$, the Moreau decomposition [Mor62] allows us to simplify (4.13) as

$$\nu \|A(x)\| \leq \text{dist}\left(\mathbf{D}A(x)^\top \cdot A(x), \text{cone}(\partial g(x))\right) \tag{4.14}$$

where $\text{dist}(\cdot, \text{cone}(\partial g(x)))$ returns the Euclidean distance to $\text{cone}(\partial g(x))$.

We may think of the geometric regularity in Definition 4 as a local condition, which might hold within a neighborhood of the constraint set $\{x : A(x) = 0\}$ rather than everywhere in $\mathbb{R}^d$. To recap, let us point out that, unlike the bulk of the non-convex programming literature, we can verify the geometric regularity in Definition 4 for a number of important examples. In this work, we will focus on instances of problem (4.1) that satisfy the geometric regularity.

4.3 Linearized AL Algorithm

To solve the equivalent formulation of problem (4.1) presented in (4.3), we propose a Linearized Augmented Lagrangian algorithm (LAL), summarized in Algorithm 3. At every iteration, Algorithm 3 takes proximal gradient descent step in the primal followed by an ascent step in the dual. The increasing sequence of penalty weights $\{\beta_k\}_k$ in Step 1, and the dual updates in Steps 6 and 7 are responsible for continuously enforcing the constraints in (4.1).

As we will see in the convergence analysis, the particular choice of β_k in Algorithm 3 strikes a balance between reducing the objective of (4.1) and enforcing its constraints. In the k^{th} iteration,

if the primal step size γ_k is sufficiently small, it is natural to expect Step 3 of Algorithm 3 to reduce the objective of (4.3). Perhaps less obviously, the geometric regularity in Definition 4 ensures that this primal update also reduces the feasibility gap of (4.1). This intuition is the key to the analysis of Algorithm 3, as formalized below and proved in Section 4.15.

Algorithm 3 LAL

Input: $\beta_1, \sigma_1, \rho, \tau > 0$, $x_1 \in \mathbb{R}^d$ with $\|A(x_1)\| \le \rho$, $y_1 \in \mathbb{R}^m$.
For $k = 1, 2, \ldots$

1. $\beta_k \leftarrow \beta_1 \sqrt{k} \log(k+1)/\log 2$
2. Let $\gamma_k \leftarrow \gamma$ by (4.12) with $x = x_k, y = y_k, \beta = \beta_k$
3. $x_{k+1} \leftarrow P_{\gamma_k, g}(x_k - \gamma_k \nabla \mathscr{L}_{\beta_k}(x_k, y_k))$
4. **If** $\gamma_k \|G_{\beta_k,\gamma_k}(x_k, y_k)\|^2 + \sigma_k \|A(x_k)\|^2 \le \tau$;
   return x_{k+1}
5. $\sigma_{k+1} \leftarrow \sigma_1 \min\Big(\frac{1}{\sqrt{k+1}}, \frac{\|A(x_1)\|}{\|A(x_{k+1})\|} \cdot \frac{\log^2 2}{(k+1)\log^2(k+2)}\Big)$.
6. $y_{k+1} \leftarrow y_k + \sigma_{k+1} A(x_{k+1})$.

Lemma 4.3.1. *For integers $k_0 < k_1$, consider the integer interval $K = [k_0 : k_1]$. Suppose that problem (4.1) satisfies the geometric regularity in Definition 4 with*

$$\nu \ge 2\lambda_A \max_{k \in K} \gamma_k \|G_{\beta_k,\gamma_k}(x_k, y_k)\|, \tag{4.15}$$

where λ_A was defined in (4.2) and

- $\rho \ge \max_{k\in K} \|A(x_k)\|, \quad \rho' \ge \max_{k\in K} \|x_k\|$,
- $S \supseteq \bigcup_{k\in K} P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(\mathbf{D}A(x_{k+1})^\top A(x_{k+1})\right)$.

Then, for every $k \in K$, it holds that

$$\|A(x_k)\| \le \frac{2}{\nu\beta_k}\Big(\|G_{\beta_k,\gamma_k}(x_k, y_k)\| + \lambda'_f + \lambda'_A \|y_k\|\Big), \tag{4.16}$$

where $\lambda'_f := \max_{\|x\|\le\rho'} \|\nabla f(x)\|, \quad \lambda'_A := \max_{\|x\|\le\rho'} \|\mathbf{D}A(x)\|$.

With the aid of Lemma 4.3.1, we can derive the convergence rate of Algorithm 3 to first-order stationarity, with the proof deferred to Section 4.10. For the convergence metric, we will use a linear combination of the gradient mapping and the feasibility gap of problem (4.1), as motivated earlier by the discussion after Definition 1.

Before stating the theorem, we summarize the assumptions on the problem (4.1) in the sequel:

Assumptions:For $\rho, \rho' > 0$, let $X_{\rho,\rho'}$ be the information zone in Definition (3), we assume;
- Problem (4.1) is geometrically regular over $X_{\rho,\rho'}$;
- $\nabla f(x)$ is λ_f-Lipschitz continuous over $X_{\rho,\rho'}$;
- $DA(x)$ is λ_A-Lipschitz continuous over $X_{\rho,\rho'}$;
- $A(x)$ and $f(x)$ are twice continuously differentiable.

Theorem 4.3.2 (Convergence rate of LAL)**.** *For sufficiently large integers $k_0 < k_1$, consider the interval $K = [k_0 : k_1]$, and consider the output sequence $\{x_k, y_k\}_{k\in K}$ of Algorithm 3. Suppose that*

$$\mu := -\min(0, \inf_k f(x_k) + g(x_k) + \langle A(x_k), y_{k_0}\rangle) < \infty.$$

For $\rho' > 0$, in addition to the strong smoothness of f and A quantified in (4.2), let us define

$$\lambda'_f = \max_{\|x\|\le\rho'} \|\nabla f(x)\|, \quad \lambda'_A = \max_{\|x\|\le\rho'} \|\mathbf{D}A(x)\|, \tag{4.17}$$

to be the (restricted) Lipschitz constants of f and A, respectively.[1] *Suppose also that problem (4.1) satisfies the geometric regularity in Definition 4 with*

$$\nu \gtrsim \max\left(\lambda_A \max_{k\in K}\sqrt{\gamma_k\mu}, \frac{\lambda'_f + \lambda'_A}{\sqrt{\mu}}\right), \tag{4.18}$$

where

- $\rho \gtrsim \sqrt{\mu}, \quad \rho' \ge \max_{k\in K}\|x_k\|, \quad S \supseteq \bigcup_{k\in K} P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(\mathbf{D}A(x_{k+1})^\top A(x_{k+1})\right).$

Then the output sequence of Algorithm 3 satisfies

$$\min_{k\in K} \frac{1}{\lambda_f + \sqrt{m}\lambda_A\rho + d\lambda'^2_A} \cdot \frac{\|\widehat{G_{\beta_K,\gamma_k}}(x_k, y_{\tilde{k-1}})\|^2}{\sqrt{k_1}\log(k_1+1)} + \|A(x_k)\|^2 \lesssim \frac{1}{k_1 - k_0}\left(\frac{\lambda'^2_f + \lambda'^2_A}{\nu^2} + \mu\right), \tag{4.19}$$

where $\lesssim, \gtrsim$ above suppress the dependence on constants for the sake of clarity.

Geometric regularity.Geometric regularity in Definition 4 plays a key role in Theorem 4.3.2 by, broadly speaking, ensuring that the primal updates of Algorithm 3 reduce the feasibility gap as the penalty weight β_k grows. We will verify this condition for several examples in Section 4.9. As confirmed by (4.19), the larger ν, the more regular (4.1), and the faster Algorithm 3 would converge to stationarity. In fact, for Algorithm 3 to succeed, Theorem 4.3.2 requires ν to be sufficiently large, see (4.18). We do not know if (4.18) is necessary or rather an artifact of the proof technique, but it is naturally expected for the convergence rate to at least slow down when ν decreases, as corroborated by (4.19). The right-hand side of (4.18) also depends on the

[1]The restricted Lipschitz continuity assumption for f, A in (4.17) is mild. Indeed, we have already assumed in (4.2) that f, A are both continuously differentiable and, by compactness of the domain in (4.17), $\lambda'_f, \lambda'_A < \infty$.

largest primal step size $\max_{k\in K}\gamma_k$. Since γ_k is found by line search in Algorithm 3, we are unable to upper bound this quantity unless we make further assumptions on problem (4.1), or slightly modify the algorithm to cap primal step sizes. Indeed, in the experimental section, we experimentally showed that we can find a ν asatisfying both (4.18) and (4.13) However, recall that the augmented Lagrangian $\mathscr{L}_{\beta_k}(\cdot, y_k)$ is λ_{β_k} Lipschitz gradient in its first argument and thus typically $\gamma_k \propto 1/\lambda_{\beta_k}$, namely, $\gamma_k \propto 1/\sqrt{k}$.

Smoothness. The smoother f, A are in the sense of (4.2,4.17), the faster convergence would be, see (4.18,4.19). Indeed, as f, A becomes smoother, problem (4.1) more and more resembles a convex program, at least locally. Note, however, that it is often not straightforward to compute the local Lipschitz constants λ'_f, λ'_A of (4.17) in practice, but it is in general possible to loosely upper bound them. For us, it is necessary to work with λ'_f, λ'_A to translate any descent in the augmented Lagrangian (see Lemma 4.1.2) to reducing the feasibility gap (see Lemma 4.3.1).

Subspace S. The freedom over the choice of subspace S in Theorem 4.3.2 is meant to further strengthen the result One might simply set $S = \mathbb{R}^d$ in the first reading.

Faster rates. Assuming restricted strong convexity and smoothness for f in (4.1) and near-isometry for A, (approximate) linear convergence of Algorithm 3 to a global minimizer of problem (4.1) can be established [LC+19].

4.4 Linearized Alternating Direction Method of Multipliers

In convex optimization, whenever applicable, Alternating Direction Method of Multipliers (ADMM) [GM75, GM76, BPC+11] often outperforms the augmented Lagrangian method. In addition, ADMM often more efficiently handles non-smooth terms.

In light of the success of ADMM in convex optimization, in this section we develop and study a (linearized) ADMM variant of Algorithm 3. More specifically, consider the program

$$\min_{x,z} f(x) + g(x) + h(z) + l(z) \quad A(x) + B(z) = 0, \tag{4.20}$$

where $f, h : \mathbb{R}^d \to \mathbb{R}$ and $A, B : \mathbb{R}^d \to \mathbb{R}^d$ are smooth in the sense described later in this section. Above, $g, l : \mathbb{R}^d \to \mathbb{R}$ are proximal-friendly convex functions which might not be differentiable. For penalty weight $\beta \geq 0$, the augmented Lagrangian corresponding to problem (4.20) is

$$\mathscr{L}_\beta(x, z, y) = f(x) + h(z) + \langle A(x) + B(z), y\rangle + \frac{\beta}{2}\|A(x) + B(z)\|^2, \tag{4.21}$$

and problem (4.20) is therefore equivalent to the minimax program

$$\min_{x,z}\max_{y} \mathscr{L}_\beta(x,z,y). \tag{4.22}$$

To solve the equivalent formulation in (4.22), we propose the linearized ADMM in Algorithm 4. Most remarks about Algorithm 3 apply to Algorithm 4 as well and, in particular, note that Algorithm 4 performs two consecutive primal updates, one on x and then one on z.

Algorithm 4 Linearized ADMM

Input: $\beta_1, \sigma_1, \rho, \tau > 0$; $x_1, z_1 \in \mathbb{R}^d$ with $\|A(x_1) + B(z_1)\| \le \rho$; $y_1 \in \mathbb{R}^m$.
For $k = 1, 2, \ldots$

1. $\beta_k \leftarrow \beta_1 \sqrt{k} \log(k+1)/\log 2$
2. $\gamma_k \leftarrow \gamma$ by (4.26) with $x = x_k, z = z_k, y = y_k, \beta = \beta_k$
3. $x_{k+1} \leftarrow P_{\gamma_k, g}(x_k - \gamma_k \nabla_x \mathscr{L}_{\beta_k}(x_k, z_k, y_k))$
4. $\iota_k \leftarrow \iota$ by (4.27) with $x = x_{k+1}, z = z_k, y = y_k, \beta = \beta_k$
5. $z_{k+1} \leftarrow P_{\iota_k l}(z_k - \iota_k \nabla_z \mathscr{L}_{\beta_k}(x_{k+1}, z_k, y_k))$
6. **If** $\gamma_k \|G_{\beta_k,\gamma_k}(x_k, z_k, y_k)\|^2 + \iota_k \|G_{\beta_k,\iota_k}(x_{k+1}, z_k, y_k)\|^2 + \sigma_k \|A(x_k) + B(z_k)\|^2 \le \tau$; **return** x_{k+1}, z_{k+1}.
7. $\sigma_{k+1} \leftarrow \sigma_1 \min(\frac{1}{\sqrt{k+1}}, \frac{\|A(x_1)+B(z_1)\|}{\|A(x_{k+1})+B(z_{k+1})\|} \cdot \frac{\log^2 2}{(k+1)\log^2(k+2)})$
8. $y_{k+1} \leftarrow y_k + \sigma_{k+1}(A(x_{k+1}) + B(z_{k+1}))$.

To parse the details of Algorithm 4, we need to slightly change the gradient map in Definition 1 and the line search procedure in Lemma 4.1.3 to match the new augmented Lagrangian $\mathscr{L}_\beta$ in (4.21). More specifically, the corresponding gradient maps are defined as

$$G_{\beta,\gamma}(x,z,y) = \frac{x - x^+}{\gamma},\ H_{\beta,\iota}(x,z,y) = \frac{z - z^+}{\iota}, \tag{4.23}$$

where

$$x^+ = P_{\gamma,g}(x - \gamma \nabla_x \mathscr{L}_\beta(x,z,y)), \quad z^+ = P_{\iota,l}(z - \iota \nabla_z \mathscr{L}_\beta(x,z,y)), \tag{4.24}$$

and $\gamma, \iota > 0$ are the primal step sizes. Above, $P_{\gamma,g}$ and $P_{\iota,l}$ are the corresponding proximal operators. The line search procedure too is similar to Lemma 4.1.3 and we set

$$x^+_{\gamma'} = P_{\gamma',g}(x - \gamma' \nabla_x \mathscr{L}_\beta(x,z,y)), \quad z^+_{\iota'} = P_{\iota',l}(z - \iota' \nabla_z \mathscr{L}_\beta(x,z,y)), \tag{4.25}$$

$$\gamma := \max\Big\{\gamma' = \gamma_0\theta^i : \mathscr{L}_\beta(x^+_{\gamma'}, z, y) \le \mathscr{L}_\beta(x,z,y) + \Big\langle x^+_{\gamma'} - x, \nabla_x \mathscr{L}_\beta(x,z,y)\Big\rangle + \frac{1}{2\gamma'}\|x^+_{\gamma'} - x\|^2\Big\}, \tag{4.26}$$

$$\iota := \max\Big\{\iota' = \iota_0\theta^i : \mathscr{L}_\beta(x, z^+_{\iota'}, y) \le \mathscr{L}_\beta(x,z,y) + \big\langle z^+_{\iota'} - z, \nabla_z \mathscr{L}_\beta(x,z,y)\big\rangle + \frac{1}{2\iota'}\|z^+_{\iota'} - z\|^2\Big\}. \tag{4.27}$$

The analysis of Algorithm 4 is similar to that of Algorithm 3, involving also a similar version of Lemma 4.3.1. The convergence rate of Algorithm 4 is detailed below and proved in Section 4.11. To present the result, let us introduce some additional notation. We let

$$u = \begin{bmatrix} x^\top z^\top \end{bmatrix}^\top, \quad p(u) = f(x) + h(z), \quad q(u) = g(x) + l(z), \quad D(u) = A(x) + B(z), \tag{4.28}$$

and assume that p, D are smooth in the sense that

$$\|\nabla p(u) - \nabla p(u')\| \le \lambda_p \|u - u'\|, \quad \|\mathbf{D}D(u) - \mathbf{D}D(u')\| \le \lambda_D \|u - u'\|, \tag{4.29}$$

for every $u, u' \in \mathbb{R}^{2d}$.

Theorem 4.4.1 (Convergence rate of linearized ADMM)**.** *For sufficiently large integers $k_0 < k_1$, consider the interval $K = [k_0 : k_1]$, and consider the output sequence $\{x_k, z_k, y_k\}_{k\in K}$ of Algorithm 3 with $\kappa_k := \min(\gamma_k, \iota_k)$ for short. After recalling (4.28), suppose that*

$$\mu := -\min(0, \inf_k p(u_k) + q(u_k) + \langle D(u_k), y_{k_0}\rangle) < \infty.$$

For $\rho' > 0$, in addition to the strong smoothness of p and D quantified in (4.29), let us define

$$\lambda'_p = \max_{\|u\| \le \rho'} \|\nabla p(x)\|, \quad \lambda'_D = \max_{\|u\| \le \rho'} \|\mathbf{D}D(x)\|, \tag{4.30}$$

to be the (restricted) Lipschitz constants of p and D, respectively. Suppose also that problem (4.20) satisfies the geometric regularity in Definition 4 with

$$\nu \gtrsim \max\left(\lambda_D \max_{k\in K} \sqrt{\kappa_k \mu}, \frac{\lambda'_p + \lambda'_D}{\sqrt{\mu}}\right), \tag{4.31}$$

where

- $\rho \gtrsim \sqrt{\mu}, \qquad \rho' \ge \max_{k\in K} \|u_k\|, \quad S \supseteq \bigcup_{k\in K} P_{\mathrm{cone}(\partial q(u_{k+1}))^*}\big(\mathbf{D}D(u_{k+1})^\top D(u_{k+1})\big).$

Table 4.1 – Comparison of the complexity results of several methods in the literature to the KKT points of (4.1).

Method	type	constraint	assumption	complexity	simplicity	ADMM extension
iPPP [LMX19]	penalty	non-convex non-convex	$A(x_0)=0$ Def. 4	$\tilde{O}(\epsilon^{-4})$ $\tilde{O}(\epsilon^{-3})$	Double-loop	✗
iALM [SAL$^+$19]	AL	non-convex	Def. 4	$\tilde{O}(\epsilon^{-4})$	Double-loop	✗
improved-iALM [LCL$^+$21]	AL	convex non-convex	Def. 4 Def. 4	$\tilde{O}(\epsilon^{-\frac{5}{2}})$ $\tilde{O}(\epsilon^{-3})$	Triple-loop	✗
GDPA [Lu22]	AL	non-convex	Def 4	$\tilde{O}(\epsilon^{-3})$	Single-loop	✗
LAL (this work)	AL	non-convex	Def. 4	$\tilde{O}(\epsilon^{-4})$	Single-loop	✓

Then the output sequence of Algorithm 3 satisfies

$$\min_{k\in K} C_1 \cdot \frac{\|\widehat{G_{\beta_K,\gamma_k}}(x_k, y_{\tilde{k}-1})\|^2 + \|\widehat{H_{\beta_K,\iota_k}}(z_k, y_{\tilde{k}-1})\|^2}{\sqrt{k_1}\log(k_1+1)} + \|A(x_k)\|^2 + \|B(z_k)\|^2 \lesssim \frac{1}{k_1-k_0}\left(C_2+\mu\right), \tag{4.32}$$

where C_1, C_2 are constants; $\lesssim, \gtrsim$ suppress the dependence on constants for the sake of clarity.

4.5 Related Works

In this section, we summarize the existing works in the context of our contributions.

Constrained optimization and ALM.[LX20] studies an hybrid (AL+penalty based) method to solve 4.1 with a linear operator. They achieve $\tilde{O}(\epsilon^{-\frac{5}{2}})$ rate of convergence under Slater's condition. However, their theory does not apply when the operator is not linear. Similar approaches with convex constraints appear in [KMM19, LMX19, LX20]. [MLY20] proposes a subgradient framework for weakly convex functions with weakly convex functional constraints. Their algorithm requires the knowledge of primal domain radius, as well as the weak convexity constants of the objective and the constraints which are often not known. Hence, this introduces 3 more tuning parameters to algorithm. [LMX19] uses an inexact proximal point method and proves $\tilde{O}(\epsilon^{-4})$ complexity. They improve this rate to $\tilde{O}(\epsilon^{-3})$ with the additional regularity assumption [SAL$^+$19] introduces an inexact AL based algorithm and proves $\tilde{O}(\epsilon^{-4})$ rate of convergence under the same regularity assumption. Later, this convergence rate is improved to $\tilde{O}(\epsilon^{-3})$ in [LCL$^+$21] by using a different solver for the inner problems. All these methods has more than one loop. To our knowledge, this is the first work which uses a single loop algorithm to solve the constrained problem in 4.1. There is also a concurrent work who established $\tilde{O}(\epsilon^{-3})$ convergence rate under the same assumption on the constraint set. However, it should be noted that their method is not applicable in ADMM setting. Table 4.1 summarizes these results.

Smooth min-max optimization. There is a surge of interest in algorithms for solving min-max optimization problems of the form;

$$\min_x \max_y \phi(x, y), \tag{4.33}$$

where the function $\phi(\cdot, \cdot)$ is smooth. This template has several applications in machine learning, including generative adversarial networks [GPM+14]. When the function $\phi(x, y)$ is non-convex in x and concave in y, this template is able to handle 1.4 with $g(x) = 0$. [LJJ20a] proposes a single loop gradient descent-ascent algorithm with $\mathcal{O}(\epsilon^{-2})$ complexity to solve 4.33. The framework assumes a bounded dual domain which eliminates many important applications. Instead, we use a special step-size rule to control the growth in the dual and remove the boundedness assumption. [LJJ20b] proposes near-optimal algorithms for a broad class of optimization problems. They obtain $\mathcal{O}(\epsilon^{-2.5})$ convergence rate to the stationary point of 4.33. However, the algorithm requires to set the final accuracy in advance and has a triple loop structre. Moreover, it is not obvious if these rates would trivially translate to $\epsilon-$KKT points of problem (4.1) since the metrics to prove convergence in these algorithms are not the same as ours.

Non-convex ADMM. We note that the convergence but not the rate of a linearized and non-convex variant of ADMM for (4.1) has been studied in [LSG19], yet it is for the linearly constrained problems. Other variants of linearized ADMM in more limited settings have appeared in [BN18, HLR16]. [WZ21] extends the classical ADMM approach to solve nonlinear equality constraint problems but they do not study the convergence of the method. Hence, to our knowledge, this is the first work which studies the convergence of the ADMM for nonlinear constraints subject to a verifiable regularity condition.

4.6 Numerical evidence on k-means Clustering

We solve the BM factorization of SDP clustering problem presented in [PW07], where

$$f(x) = \sum_{i,j=1}^{n} D_{i,j} \langle x_i, x_j \rangle, \qquad g = \delta_C,$$
$$A(x) = [x_1^\top \sum_{j=1}^{n} x_j - 1, \cdots, x_n^\top \sum_{j=1}^{n} x_j - 1]^\top,$$

where $D \in \mathbb{R}^{n\times n}$ is the Euclidean distance matrix, C is the intersection of the positive orthant in $\mathbb{R}^d$ with the Euclidean ball of radius $\sqrt{s}$. , s is the number of clusters and $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones.

To be able to solve the clustering problem with Linearized ADMM, we can rewrite the same problem in the form of 4.20 and apply the Algorithm 4;

$$
\begin{aligned}
f(x) &= \sum_{i,j=1}^{n} D_{i,j} \langle x_i, x_j \rangle, \quad g = \delta_{C_1}, \\
h(z) &= 0, \quad l = \delta_{C_2} \\
B_1(z) &= [z_1^\top \sum_{j=1}^{n} z_j - 1, \cdots, z_n^\top \sum_{j=1}^{n} z_j - 1]^\top, \\
A(x) + B_2(z) &= [x_{11} - y_{11}, \cdots, x_{ij} - z_{ij}, \cdots, x_{nr} - z_{nr}]^\top, \\
&\text{for} \quad i = 1, \cdots, n \quad \text{and} \quad j = 1, \cdots, r,
\end{aligned}
$$

where C_1 is the Euclidean ball of radius $\sqrt{s}$ and C_2 is the nonnegative orthant in $\mathbb{R}^d$. We verify the regularity condition for clustering in the Section 4.9.1.

We devote this section to (i) analyze the performance of our algorithms; to (ii) empirically validate that the regularity assumption we make indeed hold in practice (see Figure 4.1) and to (iii) showcase the practicality and tuning efficiency of our methods. The result of (iii) is deferred to Section 4.9.1. We solve the non-convex Burer-Monteiro (BM) factorization of SDP clustering problem presented in [PW07]. In our simulations, we compare our two proposed algorithms LAL and Linearized ADMM against iALM in [SAL$^+$19] and improved-iALM [LCL$^+$21]. At each iteration of the iALM, we are required to solve the subproblem up to an accuracy predefined by the user. We use a first order accelerated method (apgm) for subproblems. We initialize all methods from the same near feasible point obtained by running a least squares algorithm for consistency. Then, we tune all the methods and compare them in terms of the k-means clustering value obtained by a rounding procedure. For iALM, there are 3 different tuning parameters which affects the performance of the algorithm significantly, which are β_1, b and m_0. β_1 is the initial penalty weight, b is the rate at which penalty weight increases and m_0 is the number of initial iterations for the inner solver which also increases linearly with b [SAL$^+$19]. For LAL and Linearized ADMM, on the other hand, there is only one tuning parameter which we have to tune. For Linearized ADMM, we set $\sigma_1 = 10 * \beta_1$ for each corresponding β_1. For iALM and LAL, we set $\sigma_1 = 10^2 * \beta_1$. For improved-iALM, we use the suggested values given in [LCL$^+$21]We use the same setup as explained in [SAL$^+$19] with fashion MNIST (and additionally MNIST for condition verification) datasets. We refer the readers to [SAL$^+$19] and references therein. We ran the experiments on MacBook Pro with 8-Core Intel Core i9 processor and 32 GB of RAM.

Table 4.2 compiles the experimental results and we can see that our framework is flexible in the sense that it can also solve the ADMM formulation and both of the algorithms performs similarly. The main power of our framework comes from the tuning efficiency of the methods which are depicted through Table 4.3 in the Section 4.9.1. Moreover, Figure 4.1 provides an experimental validation to the assumptions we made in the theorems, particularly (4.13) and (4.15).

Table 4.2 – Comparison of practical performance of algorithms for solving non-convex relaxation of kmeans SDP problem on fashion MNIST (fMNIST) and MNIST datasets. Primal feasibility, kmeans value after the rounding, and time.

	fMNIST			MNIST		
method	pfeas	kmeans	time(secs)	pfeas	kmeans	time (secs)
iALM [SAL+19]	5.26e-1	28.726	11.03	3.5e-2	33.42	9.12
improved-iALM [LCL+21]	4.10e-2	29.412	8.220	6.1e-3	42.53	6.93
LAL (this work)	7.53e-1	**28.726**	**6.726**	2.7e-2	33.42	**4.33**
Linearized ADMM (this work)	**3.52e-3**	**28.726**	7.08	**5.9e-4**	**32.96**	5.65

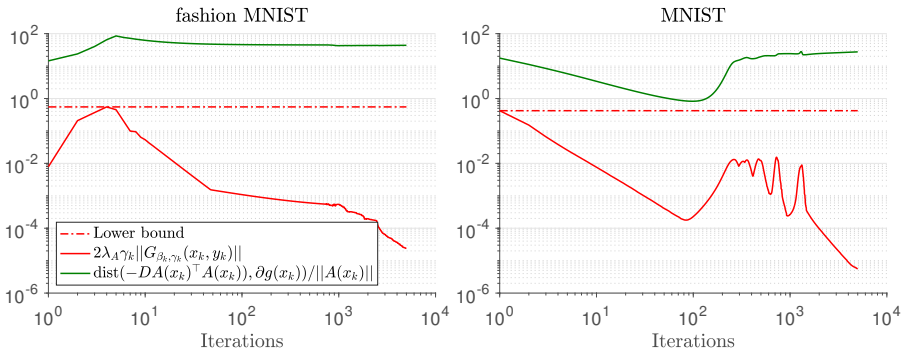

Figure 4.1 – Experimental validation showing that assumptions on the value of ν does not contradict each other. We need green line to be always larger than or equal to the maximum point of the red line which is denoted by `Lower bound`. Indeed, as the algorithm converges, the redline goes to zero, while the green line converges to a constant value. We require: $\text{dist}(-DA(x_k)^\top A(x_k)), \partial g(x_k))/||A(x_k)|| \geq \nu \geq 2\lambda_A \max_k \gamma_k ||G_{\beta_k,\gamma_k}(x_k, y_k)||$

4.7 Convergence rates

Loosely speaking, Theorem 4.3.2 states that Algorithm 3 achieves first-order stationarity for by reducing the gradient map and the feasibility gap at the rates

$$\min_{k\in K} \|\widehat{G_{\beta_k,\gamma_k}}(x_k, \tilde{y}_{k-1})\|^2 = \frac{1}{\widetilde{O}(\sqrt{k_1})}, \qquad \min_{k\in K} \|A(x_k)\| = \frac{1}{\widetilde{O}(\sqrt{k_1})}, \tag{4.34}$$

where $\widetilde{O}$ suppresses logarithmic terms above. For comparison, if f was convex and A was affine in (4.1), both rates in (4.34) would have improved to $1/k$, see [Xu17a],Theorem 2.5, for the details. Note that Theorem 4.3.2 states a stronger result than just 4.34 since we need the sum of both terms go to zero for convergence. Equation 4.34 does not guarantee this claim. Similar arguments hold for Theorem 4.4.1.

4.8 Further Discussion on Geometric Regularity

As discussed previously, we only require this regularity condition to hold near the feasible set of the problem, i.e., on the bounded information zone. Hence, we can re-write the regularity

condition on this zone as follows.

Definition 5. ***(Geometric regularity-reformulated)*** *In problem (4.1) with $m \le d$, for $\rho, \rho' > 0$ and subspace $S \subset \mathbb{R}^d$, let*

$$\nu := \begin{cases} \min\limits_{x \in \mathbb{R}^d} \frac{\|P_S P_{\mathrm{cone}(\partial g(x))^*}(\mathbf{D}A(x)^\top \cdot A(x))\|}{\|A(x)\|} \\ \textit{subject to } 0 < \|A(x)\| \le \rho, \|x\| \le \rho', \end{cases} \tag{4.35}$$

where $\nu = \nu(g, A, S, \rho, \rho')$, $\mathrm{cone}(\partial g(x))$ is the conic hull of the subdifferential $\partial g(x)$ and $P_{\mathrm{cone}(\partial g(x))^*}$ projects onto the polar of this cone, see Section 4.1.1. Moreover, $\mathbf{D}A(x)$ is the Jacobian of A. We say that (4.1) satisfies the geometric regularity if $\nu(g, A, S, \rho, \rho') > 0$.

A few remarks about geometric regularity are in order.

Polyak-Lojasiewicz (PL) inequality. The PL inequality relates the norm of the gradient of the objective function $\|\nabla \hat{f}(x)\|$ to function sub-optimality $(\hat{f}(x) - \hat{f}^\star)$ for the problem $\min_{x \in \mathbb{R}^d} \hat{f}(x)$, where $\hat{f}$ is $\lambda_{\hat{f}}$-smooth. PL inequality holds for some $\mu > 0$ if

$$\frac{1}{2}\|\nabla \hat{f}(x)\|^2 \ge \mu(\hat{f}(x) - \hat{f}^\star), \quad \forall x \in \mathbb{R}^d.$$

This is often used to obtain fast rates in non-convex optimization [KNS16]. If $g = 0$ and $S = \mathbb{R}^d$ then the regularity condition in Definition 4 reduces to PL condition for minimizing $\hat{f}(x) = \frac{1}{2}\|A(x)\|^2$ with $\nu = \sqrt{2\mu}$ [SAL$^+$19, KNS16].

Uniform regularity. [BST18] Let $A : \mathbb{R}^d \to \mathbb{R}^m$ be a continuously differentiable mapping whose Jacobian is denoted as $\mathbf{D}A(x) \in \mathbb{R}^{m \times d}$, and let Ω be a non-empty subset of $\mathbb{R}^d$. Then, A is uniformly regular on S with constant $\nu > 0$ if the following holds true:

$$\|\mathbf{D}A(x)^\top v\| \ge \nu \|v\| \qquad \forall x \in \Omega, \quad \forall v \in \mathbb{R}^m.$$

If $g = 0$ and $S = \mathbb{R}^d$, we can argue that **If uniform regularity holds, then Definition 4 holds**. In this case, uniform regularity is a stronger assumption since it requires to hold for all $v \in \mathbb{R}^m$, whereas we only require it to hold for $v = A(x), \quad \forall x \in X_{\rho,\rho'}$.

Similar connections under special cases can also be formed for other constraint qualifications such as Mangasarian-Fromovitz condition [BST18] and Kurdyka-Lojasiewicz condition [XY17, SAL$^+$19].

Jacobian $\mathbf{D}A$**.** Let $\mathbf{D}A(x)^\top \overset{\mathrm{QR}}{=} Q(x)R(x)$ be the QR decomposition of $\mathbf{D}A(x)^\top$. As we will see shortly, $\mathbf{D}A(x)^\top$ in (4.35) may be replaced with its orthonormal basis $Q(x)$ to broaden the applicability of the geometric regularity to the cases where $\mathbf{D}A(x)$ might be rank-deficient.

Definition 4 avoids this additional layer of complexity and instead, whenever needed, we assume that $\mathbf{D}A(x)$ is nonsingular for all x. Alternatively, a simple QR decomposition can also remove any redundancies from $A(x) = 0$ in the constraints of (4.1).

Subspace S. Role of the subspace S in (4.35) is also to broaden the applicability of the geometric regularity. In particular, when $S = \mathbb{R}^d$, the Moreau decomposition [Mor62] allows us to simplify (4.35) as

$$\nu := \begin{cases} \min\limits_{x} \frac{\operatorname{dist}(\mathbf{D}A(x)^\top \cdot A(x), \operatorname{cone}(\partial g(x)))}{\|A(x)\|} \\ \text{subject to } 0 < \|A(x)\| \le \rho, \|x\| \le \rho', \end{cases} \tag{4.36}$$

where $\operatorname{dist}(\cdot, \operatorname{cone}(\partial g(x)))$ returns the Euclidean distance to $\operatorname{cone}(\partial g(x))$.

Convex case. To better parse Definition 4, let us consider the specific example where $f : \mathbb{R}^d \to \mathbb{R}$ is convex, $g = 1_C$ is the indicator function for a bounded convex set $C \subset \mathbb{R}^d$, and A is a nonsingular linear operator represented with the full-rank matrix $A \in \mathbb{R}^{m \times d}$, allowing some abuse of notation. We also let $T_C(x)$ denote the tangent cone to C at x [Roc15], and reserve $P_{T_C(x)} : \mathbb{R}^d \to \mathbb{R}^d$ for the orthogonal projection onto this cone.

We can now study the interpretation of geometric regularity in this setting. Using the Moreau decomposition, it is not difficult verify that

$$\begin{aligned} \nu(1_C, A, S, \rho, \rho') &\ge \begin{cases} \min\limits_{v,x} \frac{\|P_S P_{T_C(x)} A^\top v\|}{\|v\|} \\ \text{subject to } 0 < \|v\| \le \rho, \|x\| \le \rho' \end{cases} \\ &= \begin{cases} \min\limits_{x} \eta_{\min}\left(P_S P_{T_C(x)} A^\top\right) \\ \text{subject to } \|x\| \le \rho' \end{cases} =: \widetilde{\nu}(1_C, A, S, \rho, \rho'), \end{aligned} \tag{4.37}$$

where $\eta_{\min}(\cdot)$ returns the smallest singular value of its input matrix. Then, loosely speaking, geometric regularity ensures that the row span of A is not tangent to C, similar to the Slater's condition. One might also expect ν in Definition 4 to hold some information about the convergence speed of an algorithm. For example, suppose in problem (4.1) that $f \equiv 0$, $g = 1_C$ with convex set C specified below, and A is a linear operator. We also take $m = 1$, so that A can be represented with a $1 \times d$ row-vector. For a small perturbation vector $\varepsilon \in \mathbb{R}^{1 \times d}$, let

$$C = \{x \in \mathbb{R}^d : (A + \varepsilon)x \ge 0\}$$

be a half-space. Then the Slater's condition holds regardless of the perturbation level $\|\varepsilon\|$. However, even though positive, $\nu(1_C, A, \mathbb{R}^d, \infty, \rho')$ can be made arbitrarily small by making $\|\varepsilon\|$ very small, which in turn holds a clue to the arbitrarily slow convergence of the alternating projection algorithm to solve (4.1).

To recap, let us point out that, unlike the bulk of the non-convex programming literature, we can verify the geometric regularity in Definition 4 for a number of important examples.

4.9 Additional Experiments

4.9.1 Clustering

We solve the Burer-Monteiro (BM) factorization of SDP clustering problem presented in [PW07], where

$$f(x) = \sum_{i,j=1}^{n} D_{i,j} \langle x_i, x_j \rangle, \quad g = \delta_C, \quad A(x) \qquad = [x_1^\top \sum_{j=1}^{n} x_j - 1, \cdots, x_n^\top \sum_{j=1}^{n} x_j - 1]^\top,$$

where $D \in \mathbb{R}^{n \times n}$ is the Euclidean distance matrix, C is the intersection of the positive orthant in $\mathbb{R}^d$ with the Euclidean ball of radius $\sqrt{s}$. , s is the number of clusters and $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. To be able to solve the clustering problem with Linearized ADMM, we can rewrite the same problem in the form of 4.20 and apply the Algorithm 4;

$$f(x) = \sum_{i,j=1}^{n} D_{i,j} \langle x_i, x_j \rangle, \quad g = \delta_{C_1}, \quad h(z) = 0, \quad l = \delta_{C_2}$$
$$B_1(z) = [z_1^\top \sum_{j=1}^{n} z_j - 1, \cdots, z_n^\top \sum_{j=1}^{n} z_j - 1]^\top, A(x) + B_2(z) = [x_{11} - y_{11}, \cdots, x_{nr} - z_{nr}]^\top,$$

where C_1 is the Euclidean ball of radius $\sqrt{s}$ and C_2 is the nonnegative orthant in $\mathbb{R}^d$. We verify the regularity condition for clustering in the Section 4.9.1.

Condition verification.[SAL+19] studies algorithmic-dependent variant of Definition 4 and verifies that for the clustering problem. We verify the new condition here for the same problem without algorithmic variables.

Note that,

$$A(x) = VV^\top \mathbf{1} - \mathbf{1}, \tag{4.38}$$

$$DA(x) = \begin{bmatrix} w_{1,1}x_1^\top & \cdots & w_{1,n}x_1^\top \\ \vdots & & \\ w_{n,1}x_n^\top & \cdots & w_{n,n}1x_n^\top \end{bmatrix} = \begin{bmatrix} V & \cdots & V \end{bmatrix} + \begin{bmatrix} x_1^\top & & \\ & \ddots & \\ & & x_n^\top \end{bmatrix}, \tag{4.39}$$

where $w_{i.i} = 2$ and $w_{i,j} = 1$ for $i \neq j$. In the last line above, n copies of V appear and the last matrix above is block-diagonal. For x, define V accordingly and let x_i be the ith row of V. Consequently,

$$DA(x)^\top A(x) = \begin{bmatrix} (V^\top V - I_n)V^\top \mathbf{1} \\ \vdots \\ (V^\top V - I_n)V^\top \mathbf{1} \end{bmatrix} + \begin{bmatrix} x_1(VV^\top \mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_n(VV^\top \mathbf{1} - \mathbf{1})_n \end{bmatrix}, \tag{4.40}$$

where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. Let us make a number of simplifying assumptions. First, we assume that $\|x\| < \sqrt{s}$ (which can be enforced in the iterates by replacing C with $(1-\epsilon)C$ for a small positive ϵ in the subproblems). Under this assumption, it follows that

$$(\partial g(x))_i = \begin{cases} 0 & (x)_i > 0 \\ \{a : a \leq 0\} & (x)_i = 0, \end{cases} \qquad i \leq d. \tag{4.41}$$

Second, we assume that V has nearly orthonormal columns, namely, $V^\top V \approx I_n$. This can also be enforced in each iterate of Algorithm 3 and naturally corresponds to well-separated clusters. While a more fine-tuned argument can remove these assumptions, they will help us simplify the presentation here. We also set $S = \mathbb{R}^d$ in 4.13. Under these assumptions,

$$\begin{aligned} &\operatorname{dist}\left(-DA(x)^\top A(x), \operatorname{cone}(\partial g(x))\right)^2 \\ &= \left\| \left(-DA(x)^\top A(x)\right)_+ \right\|^2 \qquad (a_+ = \max(a,0)) \\ &= \left\| \begin{bmatrix} x_1(VV^\top \mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_n(VV^\top \mathbf{1} - \mathbf{1})_n \end{bmatrix} \right\|^2 \qquad (x \in C \Rightarrow x \geq 0) \\ &= \sum_{i=1}^n \|x_i\|^2 (VV^\top \mathbf{1} - \mathbf{1})_i^2 \\ &\geq \min_i \|x_i\|^2 \cdot \sum_{i=1}^n (VV^\top \mathbf{1} - \mathbf{1})_i^2 \\ &= \min_i \|x_i\|^2 \cdot \|VV^\top \mathbf{1} - \mathbf{1}\|^2. \end{aligned} \tag{4.42}$$

Therefore, given a prescribed ν, ensuring $\min_i \|x_i\| \geq \nu$ guarantees that regularity condition holds. When the algorithm is initialized close enough to the constraint set, there is indeed no need to

Table 4.3 – K-means clustering results obtained by running each method with the corresponding set of hyper-parameters ($\{\beta_1, b, m_0\}$ for iALM and only $\{\beta_1\}$ for Linearized ADMM) for 1000 iterations for the fashionMNIST dataset and rounding afterwards. Note that Linearized ADMM and LAL both have only 1 tuning parameter and can perform at least as good as iALM. Moreover, tuning iALM takes exponentially longer as the tuning interval increases than tuning Linearized ADMM and LAL. This table is to show that Linearized ADMM and LAL are both tuning efficient methods.

m_0	b	β_1	iALM [SAL$^+$19]	Linearized ADMM	LAL
10	1.1	10^{-2}	87.89044	794.98999	28.73240
		10^{-1}	28.73417	688.78644	28.73418
		1	86.64399	28.72636	28.72636
		10^{1}	28.76799	93.45923	93.45923
		10^{2}	28.72689	159.66110	159.66110
	2	10^{-2}	113.44376		
		10^{-1}	28.73471		
		1	45.25990		
		10^{1}	28.72638		
		10^{2}	28.72689		
	4	10^{-2}	109.74625		
		10^{-1}	28.77593		
		1	42.90821		
		10^{1}	65.02499		
		10^{2}	28.73470		
	8	10^{-2}	80.58079		
		10^{-1}	45.81465		
		1	43.33508		
		10^{1}	28.72690		
		10^{2}	28.86121		
50	1.1	10^{-2}	93.18016		
		10^{-1}	33.23707		
		1	28.79125		
		10^{1}	28.72691		
		10^{2}	28.79532		
	2	10^{-2}	93.18016		
		10^{-1}	29.41133		
		1	28.79125		
		10^{1}	28.72689		
		10^{2}	28.72689		
	4	10^{-2}	480.24234		
		10^{-1}	28.72690		
		1	28.79125		
		10^{1}	28.72689		
		10^{2}	28.72689		
	8	10^{-2}	103.52721		
		10^{-1}	28.72690		
		1	28.79125		
		10^{1}	28.72689		
		10^{2}	28.72689		

separately enforce (4.42). In practice, often n exceeds the number of true clusters and a more intricate analysis is required to establish the inequality by restricting the argument to a particular subspace of $\mathbb{R}^n$.

Practicality of the approach. Table 4.3 compiles the experimental results for solving clustering problem with Linearized ADMM and LAL. Note that Linearized ADMM and LAL both have only 1 tuning parameter and can perform at least as good as iALM with significantly less tuning time. Moreover, tuning iALM takes exponentially longer as the tuning parameters increase. iALM is highly reluctant to changes in the set of tuning parameters $\{\beta_1, b, m_0\}$ as depicted in Table 4.3.

4.9.2 Maxcut

In this section, we show that regularity condition in Definition 4 holds for the maxcut problem. Given an undirected graph $G = (V, E)$, maximum cut problem aims to find the set of vertices which maximizes the weight of the edges in the cut. Here, V denotes the set of vertices, and E denotes the edges. It is an important combinatorial problem in computer science and NP-Hard. We solve the following SDP relaxation of the problem;

$$\min_{X \in \mathbb{R}^{n \times n}} \quad \frac{1}{4}\langle C, X\rangle \quad \text{s.t.} \quad \operatorname{diag}(X) = \mathbf{1}, \quad \operatorname{trace}(X) = n, \quad X \succeq 0, \tag{4.43}$$

where $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones, $C \in \mathbb{R}^{n \times n}$ is the symmetric graph Laplacian matrix. We solve the following BM factorization of the problem after change of variables $X = YY^\top$;

$$\min_{Y \in \mathbb{R}^{n \times r}} \quad \frac{1}{4}\operatorname{trace}(Y^\top C Y) \quad \text{s.t.} \quad \operatorname{diag}(YY^\top) = \mathbf{1}, \quad \|Y\|_F^2 \leq n. \tag{4.44}$$

Note here that we relaxed the equality constraint $\operatorname{trace}(X) = n$ to an inequality constraint $\|Y\|_F^2 < n$ while doing the factorization. Although the constraints are not equivalent, at the feasible point of the problem, $\|Y\|_F^2 = n$ is automatically satisfied. For every i, let $x_i \in \mathbb{R}^r$ denote the ith row of Y. Let us form $x \in \mathbb{R}^d$ with $d = nr$ by vectorizing Y, namely,

$$x = [x_1^\top \cdots x_n^\top]^\top. \tag{4.45}$$

We can therefore cast the above in the form of the original problem with

$$f(x) = \sum_{i,j} C_{i,j}\langle x_i, x_j\rangle, \quad A : x \to [\|x_1\|^2 - 1, \cdots, \|x_n\|^2 - 1]^\top. \tag{4.46}$$

Condition verification. First, calculate the jacobian of the nonlinear operator;

$$DA(x) = \begin{bmatrix} x_1^\top & \cdots & 0 \\ \vdots & & \\ 0 & \cdots & x_n^\top \end{bmatrix} \in \mathbb{R}^{d' \times d}. \tag{4.47}$$

In particular, if we take $S = \mathbb{R}^d$ and $\rho < 1$, we have $P_{\text{cone}(\partial g(x))^*} = I_d$ and thus

$$\begin{aligned} \nu(g, A, S, \rho, \rho') &= \begin{cases} \min_x \eta_{\min}(DA(x)) \\ \text{subject to } 0 < \|A(x)\| \le \rho, \|x\| \le \rho' \end{cases} \\ &= \begin{cases} \min_x \min_i \|x_i\| \\ \text{subject to } \|x\| \le \rho', \quad \sum_i |\|x_i\|^2 - 1| > 0, \quad |\|x_i\|^2 - 1| \le \rho \qquad \forall i. \end{cases} \\ &\ge \sqrt{1-\rho} > 0. \end{aligned} \tag{4.48}$$

Above, $\eta_{\min}(.)$ returns the smallest singular value of its input. Consequently, condition in (4.35) holds for the max-cut problem.

4.9.3 Continuous Optimization for Learning Structures

Learning direct acyclic graphs(DAGs) is an NP-hard problem because of the difficulty in enforcing combinatorial acyclicity constraints [ZARX18]. DAGs have many application ranging from biology [SPP+05] and genetics [ZGB+13] to machine learning [KF09] and causal inference [SGSH00]. [ZARX18] proposed a new method for solving problem of learning DAGs by converting the combinatorial problem into a nonlinearly constrained optimization problem which perfectly fits to our template. The resulting optimization problem is as follows:

$$\min_{X \in \mathbb{R}^{d \times d}} F(X) := \frac{1}{2n} \|W - WX\|_F^2 + \lambda \|X\|_1 \qquad \text{s.t.} \qquad h(X) = 0, \tag{4.49}$$

where $W \in \mathbb{R}^{n \times d}$ is a data matrix consisting n i.i.d. observations of random vector, $\|X\|_1 = \|\text{vec}(X)\|_1$ is the ℓ_1-regularization for inducing sparsity and $h(X) = \text{tr}(e^{X \circ X}) - d = 0$ is the constraint for characterizing acyclicity. In $h(X)$, $\circ$ denotes the Hadamard product and e^X is the matrix exponential of X. NOTEARS algorithm in [ZARX18] forms the augmented Lagrangian for the above problem and solves a sequence of complicated subproblems with LBFGS followed by a dual update. Our generic Algorithm LAL does one simple proximal gradient step followed by a dual ascent and it can learn the graph with a quality as high as NOTEARS. The learned adjacency matrices and resulting directed graph with LAL and NOTEARS algorithm on a 20−node graph with 1000 samples are depicted in Figures 4.2 and 4.3. Thickness of the arrow increases as the weight of the corresponding node increases. We have used public implementation of NOTEARS available at https://github.com/xunzheng/notears and implemented our method on this package.

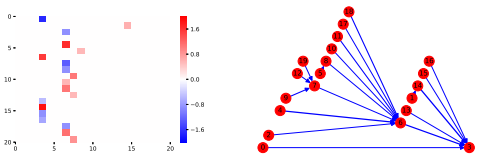

Figure 4.2 – LAL

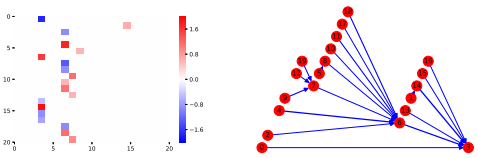

Figure 4.3 – NOTEARS

4.10 Proof of Theorem 4.3.2

For the reader's convenience, let us recall the updates of Algorithm 3 in iteration k:

$$\begin{aligned} x_{k+1} &= P_g(x_k - \gamma_k \nabla_x \mathscr{L}_{\beta_k}(x_k, y_k)) \\ &= P_g\Big(x_k - \gamma_k \nabla f(x_k) \\ &\qquad - \gamma_k \mathbf{D} A(x_k)^\top \big(y_k + \beta_k A(x_k)\big)\Big), \qquad \text{(see (4.3))} \end{aligned} \tag{4.50}$$

$$y_{k+1} = y_k + \sigma_{k+1} A(x_{k+1}). \tag{4.51}$$

Moreover, we will use the shorthand

$$G_k = G_{\beta_k,\gamma_k}(x_k, y_k) = \frac{x_k - x_{k+1}}{\gamma_k}, \qquad \text{(see (1.12))} \tag{4.52}$$

for gradient mapping throughout the proof. For integers $k_0 \le k_1$, consider the interval

$$K = [k_0 : k_1] = \{k_0, \cdots, k_1\}. \tag{4.53}$$

Since the primal step size γ_k is determined by the line search subroutine in Lemma 4.1.3, we may now apply Lemma 4.1.2 for every iteration in the interval K to find that

$$\begin{aligned} \frac{\gamma_k \|G_k\|^2}{2} &\le \mathscr{L}_{\beta_k}(x_k, y_k) + g(x_k) - \mathscr{L}_{\beta_k}(x_{k+1}, y_k) - g(x_{k+1}) \qquad \text{(see Lemma 4.1.2)} \\ &= f(x_k) + g(x_k) - f(x_{k+1}) - g(x_{k+1}) + \langle A(x_k) - A(x_{k+1}), y_k \rangle \\ &\quad + \frac{\beta_k}{2}(\|A(x_k)\|^2 - \|A(x_{k+1})\|^2), \qquad \text{(see (4.3))} \end{aligned} \tag{4.54}$$

for every $k \in K$. On the other hand, note that

$$y_k = y_{k_0-1} + \sum_{i=k_0}^{k} \sigma_i A(x_i), \qquad \text{(see (4.51))} \tag{4.55}$$

which, after substituting in (4.54), yields that

$$\begin{aligned}\frac{\gamma_k\|G_k\|^2}{2} \leq\ & f(x_k)+g(x_k)-f(x_{k+1})-g(x_{k+1})\\ &+\left\langle A(x_k)-A(x_{k+1}), y_{k_0}+\sum_{i=k_0+1}^{k}\sigma_i A(x_i)\right\rangle\\ &+\frac{\beta_k}{2}(\|A(x_k)\|^2-\|A(x_{k+1})\|^2).\end{aligned} \tag{4.56}$$

By summing up (4.56) over k from k_0 to k_1, we argue that

$$\begin{aligned}&\sum_{k=k_0}^{k_1}\frac{\gamma_k\|G_k\|^2}{2}\\ &\leq f(x_{k_0})+g(x_{k_0})-f(x_{k_1+1})-g(x_{k_1+1})+\langle A(x_{k_0})-A(x_{k_1+1}), y_{k_0}\rangle\\ &\quad+\sum_{k=k_0}^{k_1}\sum_{i=k_0+1}^{k}\sigma_i\langle A(x_k)-A(x_{k+1}), A(x_i)\rangle\\ &\quad+\sum_{k=k_0}^{k_1}\frac{\beta_k}{2}\|A(x_k)\|^2-\sum_{k=k_0}^{k_1}\frac{\beta_k}{2}\|A(x_{k+1})\|^2 \qquad \text{(see (4.56))}\\ &= f(x_{k_0})+g(x_{k_0})-f(x_{k_1+1})-g(x_{k_1+1})+\langle A(x_{k_0})-A(x_{k_1+1}), y_{k_0}\rangle\\ &\quad+\sum_{k=k_0}^{k_1}\sum_{i=k_0+1}^{k}\sigma_i\langle A(x_k)-A(x_{k+1}), A(x_i)\rangle\\ &\quad+\sum_{k=k_0}^{k_1}\frac{\beta_k}{2}\|A(x_k)\|^2-\sum_{k=k_0+1}^{k_1+1}\frac{\beta_{k-1}}{2}\|A(x_k)\|^2.\end{aligned} \tag{4.57}$$

By manipulating the last line above, we find that

$$
\begin{aligned}
&\sum_{k=k_0}^{k_1} \frac{\gamma_k \|G_k\|^2}{2} \\
&\leq f(x_{k_0}) + g(x_{k_0}) - f(x_{k_1+1}) - g(x_{k_1+1}) + \langle A(x_{k_0}) - A(x_{k_1+1}), y_{k_0} \rangle \\
&\quad + \frac{\beta_{k_0}}{2} \|A(x_{k_0})\|^2 + \sum_{i=k_0+1}^{k_1} \sum_{k=i}^{k_1} \sigma_i \langle A(x_k) - A(x_{k+1}), A(x_i) \rangle \\
&\quad + \sum_{k=k_0+1}^{k_1} \frac{\beta_k - \beta_{k-1}}{2} \|A(x_k)\|^2 - \frac{\beta_{k_1}}{2} \|A(x_{k_1+1})\|^2 \\
&\leq \mu + \sum_{i=k_0+1}^{k_1} \sigma_i \left\langle A(x_i) - A(x_{k_1+1}), A(x_i) \right\rangle \\
&\quad + \sum_{k=k_0+1}^{k_1} \frac{\beta_k - \beta_{k-1}}{2} \|A(x_k)\|^2 - \frac{\beta_{k_1}}{2} \|A(x_{k_1+1})\|^2 \qquad \text{(see (4.59))} \\
&= \mu + \sum_{k=k_0+1}^{k_1} \left(\sigma_k + \frac{\beta_k - \beta_{k-1}}{2} \right) \|A(x_k)\|^2 \\
&\quad - \sum_{k=k_0+1}^{k_1} \sigma_k \left\langle A(x_{k_1+1}), A(x_k) \right\rangle - \frac{\beta_{k_1}}{2} \|A(x_{k_1+1})\|^2, \qquad (4.58)
\end{aligned}
$$

where we assumed that

$$
\begin{aligned}
\mu := \max\Bigg(&\sup_k \Big(f(x_{k_0}) + g(x_{k_0}) - f(x_k) - g(x_k) + \langle A(x_{k_0}) - A(x_k), y_{k_0} \rangle \\
&+ \frac{\beta_{k_0}}{2} \|A(x_{k_0})\|^2 \Big), 0 \Bigg) < \infty, \qquad (4.59)
\end{aligned}
$$

Given initial step sizes $\beta_{k_0}, \sigma_{k_0} > 0$, recall that the penalty weights and the dual step sizes of Algorithm 3 are set to

$$
\beta_k = \beta_{k_0} \sqrt{\frac{k \log^2(k+1)}{k_0 \log^2(k_0+1)}},
$$

$$
\sigma_k = \sigma_{k_0} \min\left(\sqrt{\frac{k_0}{k}}, \frac{\|A(x_{k_0})\| k_0 \log^2(k_0+1)}{\|A(x_k)\| k \log^2(k+1)} \right), \qquad \forall k \in K. \qquad (4.60)
$$

For future reference, (4.60) implies that

$$\begin{aligned}
\beta_k - \beta_{k-1} &= \beta_{k-1}\left(\sqrt{\frac{k\log^2(k+1)}{(k-1)\log^2 k}} - 1\right) \\
&\le \beta_{k-1}\cdot\frac{k\log^2(k+1) - (k-1)\log^2 k}{(k-1)\log^2 k} \\
&\le \beta_{k-1}\left(\frac{k\log^2(1+\frac{1}{k})}{(k-1)\log^2 k} + \frac{1}{k-1}\right) \\
&\le \frac{2\beta_{k-1}}{k-1} \qquad (k_0 \gtrsim 1) \\
&\le \frac{2\beta_{k_0}}{k-1}\sqrt{\frac{(k-1)\log^2 k}{k_0\log^2(k_0+1)}} \qquad \text{(see (4.60))} \\
&= \frac{2\beta_{k_0}\log k}{\sqrt{(k-1)k_0}\log(k_0+1)}, \qquad \forall k \in K, \qquad (4.61)
\end{aligned}$$

when k_0 is sufficiently large. We can therefore further simplify the last line of (4.58) as

$$\begin{aligned}
&\sum_{k=k_0}^{k_1}\frac{\gamma_k\|G_k\|^2}{2} \\
&\le \mu + \sum_{k=k_0}^{k_1}\left(\sigma_k + \frac{\beta_k - \beta_{k-1}}{2}\right)\|A(x_k)\|^2 \\
&\qquad + \sum_{k=k_0}^{k_1}\sigma_k\|A(x_{k_1+1})\|\|A(x_k)\| - \frac{\beta_{k_1}}{2}\|A(x_{k_1+1})\|^2 \qquad \text{(see (4.58))} \\
&\le \mu + \sum_{k=k_0}^{k_1}\left(\sigma_k + \frac{\beta_k - \beta_{k-1} + 1}{2}\right)\|A(x_k)\|^2 \\
&\qquad + \frac{1}{2}\left(\sum_{k=k_0}^{k_1}\sigma_k^2 - \beta_{k_1}\right)\|A(x_{k_1+1})\|^2 \qquad (2ab \le a^2 + b^2) \\
&\le \mu + 2\sum_{k=k_0}^{k_1}\|A(x_k)\|^2, \qquad \text{(see (4.60,4.61))} \qquad (4.62)
\end{aligned}$$

for sufficiently large k_0. Indeed, the coefficient of $\|A(x_{k_1+1})\|$ in the second-to-last line of (4.62) is negative because

$$\begin{aligned}
&\sum_{k=k_0}^{k_1} \sigma_k^2 - \beta_{k_1} \\
&\leq \sum_{k=k_0}^{k_1} \frac{\sigma_{k_0}^2 k_0}{k} - \beta_{k_0} \sqrt{\frac{k_1 \log^2(k_1+1)}{k_0 \log^2(k_0+1)}} \qquad \text{(see (4.60))} \\
&\leq 2\sigma_{k_0}^2 k_0 \int_{k_0}^{k_1} \frac{da}{a} - \beta_{k_0} \sqrt{\frac{k_1 \log^2(k_1+1)}{k_0 \log^2(k_0+1)}} \\
&\leq 0, \qquad (4.63)
\end{aligned}$$

when k_0 is sufficiently large. Note that (4.62) bounds the gradient mapping with the feasibility gap. A converse is given by Lemma 4.3.1. In order to apply this result, let us assume that the assumptions in Lemma 4.3.1 are met. Lemma 4.3.1 is then in force and we may now substitute (4.16) back into (4.62) to find that

$$\begin{aligned}
&\sum_{k=k_0}^{k_1} \gamma_k \|G_k\|^2 \\
&\leq 2 \sum_{k=k_0}^{k_1} \|A(x_k)\|^2 + 2\mu \qquad \text{(see (4.62))} \\
&\leq 2 \sum_{k=k_0}^{k_1} \left(\frac{2}{\nu \beta_k} \left(\|G_k\| + \lambda_f' + \lambda_A' \|y_k\| \right) \right)^2 + 2\mu \qquad \text{(see (4.16))} \\
&\leq \sum_{k=k_0}^{k_1} \frac{24\|G_k\|^2}{\nu^2 \beta_k^2} + \sum_{k=k_0}^{k_1} \frac{24\lambda_f'^2}{\nu^2 \beta_k^2} + \sum_{k=k_0}^{k_1} \frac{24\lambda_A'^2 \|y_k\|^2}{\nu^2 \beta_k^2} + 2\mu, \qquad (4.64)
\end{aligned}$$

where we used the shorthand $\nu = \nu(g, A, S, \rho, \rho')$ and the last line above uses the inequality

$$\left(\sum_{i=1}^{p} a_i \right)^2 \leq p \sum_{i=1}^{p} a_i^2, \qquad (4.65)$$

for integer p and scalars $\{a_i\}_i$. If we set

$$B_K = \sum_{k=k_0}^{k_1} \frac{\|y_k\|^2}{k \log^2(k+1)}, \qquad c \geq \sum_{k=1}^{\infty} \frac{1}{k \log^2(k+1)}, \qquad (4.66)$$

and, after recalling the choice of $\{\beta_k\}_k$ in (4.60), the last line of (4.64) can be simplified as

$$\begin{aligned}\sum_{k=k_0}^{k_1}\gamma_k\|G_k\|^2 &\le 2\sum_{k=k_0}^{k_1}\|A(x_k)\|^2+2\mu\\ &\le \sum_{k=k_0}^{k_1}\frac{24\|G_k\|^2k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}+\sum_{k=k_0}^{k_1}\frac{24\lambda_f'^2k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}\\ &\quad+\sum_{k=k_0}^{k_1}\frac{24\lambda_A'^2k_0\log^2(k_0+1)\|y_k\|^2}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}+2\mu \qquad \text{(see (4.60,4.64))}\\ &\le \sum_{k=k_0}^{k_1}\frac{24\|G_k\|^2k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}+\frac{24k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(c\lambda_f'^2+\lambda_A'^2B_K\right)\\ &\quad+2\mu. \qquad \text{(see (4.66))}\end{aligned} \tag{4.67}$$

To simplify the above bound, let us assume that

$$\frac{24k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}\le\frac{\gamma_k}{2},\qquad\forall k\in K. \tag{4.68}$$

After rearranging (4.67) and applying (4.68), we arrive at

$$\begin{aligned}&\sum_{k=k_0}^{k_1}\frac{\gamma_k}{2}\|G_k\|^2\\ &\le\sum_{k=k_0}^{k_1}\left(\gamma_k-\frac{24k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2\,k\log^2(k+1)}\right)\|G_k\|^2 \qquad \text{(see (4.68))}\\ &\le\frac{24k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(c\lambda_f'^2+\lambda_A'^2B_K\right)+2\mu. \qquad \text{(see (4.67))}\end{aligned} \tag{4.69}$$

In turn, the bound above on the gradient mapping controls the feasibility gap, namely,

$$\begin{aligned}\sum_{k=k_0}^{k_1}\|A(x_k)\|^2&\le\sum_{k=k_0}^{k_1}\frac{\gamma_k\|G_k\|^2}{4}+\frac{12k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(c\lambda_f'^2+\lambda_A'^2B_K\right) \qquad \text{(see (4.67,4.68))}\\ &\le\frac{24k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(c\lambda_f'^2+\lambda_A'^2B_K\right)+\mu. \qquad \text{(see (4.69))}\end{aligned} \tag{4.70}$$

By adding (4.69,4.70), we find that

$$\sum_{k=k_0}^{k_1}\gamma_k\|G_k\|^2+\|A(x_k)\|^2\le\frac{72k_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(c\lambda_f'^2+\lambda_A'^2B_K\right)+5\mu. \tag{4.71}$$

In order to interpret (4.71), we next estimate B_K, defined in (4.66). To that end, let us first control the growth of the dual sequence $\{y_k\}_k$. Recalling (4.51) and for every $k \in K$, we write that

$$\begin{aligned}
\|y_k\| &\leq \|y_{k_0}\| + \sum_{i=k_0+1}^{k} \sigma_i \|A(x_i)\| \qquad \text{(see (4.51))} \\
&\leq \|y_{k_0}\| + \sum_{i=k_0+1}^{k} \frac{\rho\sigma_{k_0} k_0 \log^2(k_0+1)}{k\log^2(k+1)} \qquad \text{(see Lemma 4.3.1 and (4.60))} \\
&\leq \|y_{k_0}\| + c\rho\sigma_{k_0} k_0 \log^2(k_0+1) \\
&=: y_{\max}.
\end{aligned} \tag{4.72}$$

Having uncovered the growth of the dual sequence above, we evaluate B_K as

$$\begin{aligned}
B_K &= \sum_{k=k_0}^{k_1} \frac{\|y_k\|^2}{k\log^2(k+1)} \qquad \text{(see (4.66))} \\
&\leq \sum_{k=k_0}^{k_1} \frac{y_{\max}^2}{k\log^2(k+1)} \qquad \text{(see (4.72))} \\
&\leq c y_{\max}^2. \qquad \text{(see (4.66))}
\end{aligned} \tag{4.73}$$

In order to interpret (4.71), it still remains to control the primal step sizes $\{\gamma_k\}_k$. To invoke $\gamma \geq \frac{\theta}{\lambda_\beta}$, we first need to gauge how smooth the augmented Lagrangian $\mathscr{L}_{\beta_k}(\cdot, y_k)$ is as a function of its first argument. For every $k \in K$, note that

$$\begin{aligned}
\lambda_{\beta_k} &\leq \lambda_f + \sqrt{m}\lambda_A\big(\|y_k\| + \beta_k\rho\big) + \beta_k d\lambda_A'^2 \qquad \text{(see (4.10))} \\
&\leq (\lambda_f + \sqrt{m}\lambda_A y_{\max}) + \beta_k\big(\sqrt{m}\lambda_A\rho + d\lambda_A'^2\big). \qquad \text{(see (4.72))}
\end{aligned} \tag{4.74}$$

We are now in position to invoke $\gamma \geq \frac{\theta}{\lambda_\beta}$ (see Lemma 4.1.3) by writing that

$$\begin{aligned}
\gamma_k &\geq \frac{\theta}{\lambda_{\beta_k}} \\
&\geq \frac{\theta}{(\lambda_f + \sqrt{m}\lambda_A y_{\max}) + \beta_k\big(\sqrt{m}\lambda_A\rho + d\lambda_A'^2\big)} \qquad \text{(see (4.74))} \\
&\geq \frac{\theta}{2\beta_k\big(\lambda_f + \sqrt{m}\lambda_A\rho + d\lambda_A'^2\big)} \qquad \text{((4.60) and } k_0 \gtrsim 1) \\
&\geq \frac{\theta}{2\beta_{k_0}\big(\lambda_f + \sqrt{m}\lambda_A\rho + d\lambda_A'^2\big)}\sqrt{\frac{k_0\log^2(k_0+1)}{k\log^2(k+1)}} \qquad \text{(see (4.60))} \\
&=: \overline{\gamma}\sqrt{\frac{k_0\log^2(k_0+1)}{k\log^2(k+1)}},
\end{aligned} \tag{4.75}$$

for every $k \in K$. The first consequence of (4.75) is that (4.68) holds automatically when k_0 is sufficiently large. Having estimated B_K and $\{\gamma_k\}_k$, we can also rewrite (4.71). Indeed,

(4.71,4.73,4.75) together imply that

$$\sum_{k=k_0}^{k_1} \overline{\gamma}\|G_k\|^2 \sqrt{\frac{k_0 \log^2(k_0+1)}{k \log^2(k+1)}} + \|A(x_k)\|^2 \\ \leq \frac{72ck_0 \log^2(k_0+1)}{\nu^2 \beta_{k_0}^2} \left(\lambda_f'^2 + \lambda_A'^2 y_{\max}^2\right) + 5\mu, \tag{4.76}$$

and, consequently,

$$\min_{k\in K} \overline{\gamma}\|G_k\|^2 \sqrt{\frac{k_0 \log^2(k_0+1)}{k_1 \log^2(k_1+1)}} + \|A(x_k)\|^2 \\ \leq \frac{72ck_0 \log^2(k_0+1)}{\nu^2 \beta_{k_0}^2 (k_1-k_0)} \left(\lambda_f'^2 + \lambda_A'^2 y_{\max}^2\right) + \frac{5\mu}{k_1-k_0}. \tag{4.77}$$

Now we are in a position to show that $G_{\beta,\gamma}(x_{k-1}, y_{k-1})$ approaches asymptotically to $\widehat{G_{\beta,\gamma}}(x_k, y_{k-1}^{\tilde{}})$. Recall the definition of the displaced gradient mapping.

$$\widehat{G_{\beta,\gamma}}(x_k, y_{k-1}^{\tilde{}}) := G_{\beta,\gamma}(x_{k-1}, y_{k-1}) + \nabla f(x_k) - \nabla f(x_{k-1}) + (\mathbf{D}A(x_k) - \mathbf{D}A(x_{k-1}))^\top y_{k-1}^{\tilde{}}, \tag{4.78}$$

$$\leq G_{\beta,\gamma}(x_{k-1}, y_{k-1}) + \lambda_f \|x_k - x_{k-1}\| + \lambda_A \|x_k - x_{k-1}\| \|y_{k-1}^{\tilde{}}\| \tag{4.79}$$

$$\leq G_{\beta,\gamma}(x_{k-1}, y_{k-1}) + \lambda_f \gamma_{k-1} G_{\beta,\gamma}(x_{k-1}, y_{k-1}) + \lambda_A \gamma_{k-1} G_{\beta,\gamma}(x_{k-1}, y_{k-1}) \left(\|y_{\max}\| + \beta_{k-1}\rho\right) \tag{4.80}$$

Hence, we have the result since $\gamma_{k-1} \leq 1/\lambda_{\beta_{k-1}}$. We use the smoothness of $f(\cdot)$ and $\mathbf{D}A(\cdot)$, bounds we derived for y_k and boundedness of $A(x)$.

When we applied Lemma 4.3.1 earlier, we did not check whether the assumptions on ρ therein hold. Let us revisit this assumption. We first derive a weaker but uniform bound on the feasibility gap. For every $k \in K$, it holds that

$$\|A(x_k)\|^2 \leq \sum_{i=k_0}^{k_1} \|A(x_i)\|^2 \\ \leq \frac{24k_0 \log^2(k_0+1)}{\nu^2 \beta_{k_0}^2} \left(c\lambda_f'^2 + \lambda_A'^2 B_K\right) + \mu \qquad \text{(see (4.70))} \\ \leq \frac{24ck_0 \log^2(k_0+1)}{\nu^2 \beta_{k_0}^2} \left(\lambda_f'^2 + \lambda_A'^2 y_{\max}^2\right) + \mu. \qquad \text{(see (4.73))} \tag{4.81}$$

Therefore, we may replace the assumption on ρ in Lemma 4.3.1 with the stronger assumption that

$$\rho^2 \geq \frac{24ck_0 \log^2(k_0+1)}{\nu^2 \beta_{k_0}^2} \left(\lambda_f'^2 + \lambda_A'^2 y_{\max}^2\right) + \mu, \tag{4.82}$$

which, after rearranging, can be presented as

$$\nu^2 \geq \frac{24ck_0\log^2(k_0+1)}{\beta_{k_0}^2(\rho^2-\mu)}\left(\lambda_f'^2+\lambda_A'^2 y_{\max}^2\right), \qquad \rho > \sqrt{\mu}. \tag{4.83}$$

Note that, for (4.83) to hold, it is in particular necessary that $\|A(x_{k_0})\| \leq \rho\sqrt{2/\beta_{k_0}}$, as seen in (4.59). That is, for Algorithm 3 to success, it must be initialized close enough to the feasible set.

Lastly, let us revisit the lower bound on ν in (4.15) of Lemma 4.3.1. First we derive a weaker but uniform bound on the gradient mapping. For every $k \in K$, it holds that

$$\begin{aligned}
&\max_{k\in K}\gamma_k\|G_k\| \\
&\leq \max_{k\in K}\sqrt{\gamma_k}\cdot\sqrt{\max_{k\in K}\gamma_k\|G_k\|^2} \\
&\leq \max_{k\in K}\sqrt{\gamma_k}\cdot\sqrt{\sum_{k=k_0}^{k_1}\gamma_k\|G_k\|^2} \\
&\leq \max_{k\in K}\sqrt{\gamma_k}\cdot\left(\frac{48ck_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(\lambda_f'^2+\lambda_A'^2 y_{\max}^2\right)+4\mu\right)^{\frac{1}{2}}. \qquad \text{(see (4.69,4.73))}
\end{aligned} \tag{4.84}$$

Instead of (4.15), it therefore suffices to make the stronger assumption that

$$\nu \geq 2\lambda_A\max_{k\in K}\sqrt{\gamma_k}\cdot\left(\frac{48ck_0\log^2(k_0+1)}{\nu^2\beta_{k_0}^2}\left(\lambda_f'^2+\lambda_A'^2 y_{\max}^2\right)+4\mu\right)^{\frac{1}{2}}, \tag{4.85}$$

which can in turn be replaced with the stronger assumptions

$$\nu \geq \max\left(4\sqrt{2}\lambda_A\max_{k\in K}\sqrt{\gamma_k\mu}, \frac{4\sqrt{ck_0}\log(k_0+1)}{\beta_{k_0}\sqrt{\mu}}(\lambda_f'+\lambda_A' y_{\max})\right), \tag{4.86}$$

where we used the inequality $\sqrt{a+b} \leq 2\max(\sqrt{a},\sqrt{b})$ above. This completes the proof of Theorem 4.3.2.

4.11 Proof of Theorem 4.4.1

For completeness, let us repeat the technical lemmas and definitions of Section 4.1.1, slightly adjusted here for the augmented Lagrangian of problem (4.22), defined in (4.21). These standard results are stated below without proof.

Lemma 4.11.1 (Smoothness)**.** *Given $\rho, \rho' \geq 0$, it holds that*

$$\|\nabla_x\mathscr{L}_\beta(x,z,y) - \nabla_x\mathscr{L}_\beta(x',z,y)\| \leq \lambda_{\beta,z}\|x-x'\|,$$

$$\|\nabla_z \mathscr{L}_\beta(x,z,y) - \nabla_z \mathscr{L}_\beta(x,z',y)\| \le \lambda_{\beta,x}\|z-z'\|, \tag{4.87}$$

for every $(x,z),(x',z),(x,z') \in X_{\rho,\rho'}$ *and* $y \in \mathbb{R}^m$*, where*

$$X_{\rho,\rho'} := \{(x'',z'') : \|A(x'')+B(z'')\| \le \rho, \|x''\| \le \rho', \|z''\| \le \rho'\}, \tag{4.88}$$

$$\lambda_{\beta,x} \le \lambda_f + \sqrt{m}\lambda_A\big(\|y\|+\beta\rho\big) + \beta d \lambda_A'^2,$$

$$\lambda_{\beta,z} \le \lambda_h + \sqrt{m}\lambda_B\big(\|y\|+\beta\rho\big) + \beta d \lambda_B'^2, \tag{4.89}$$

$$\lambda_A' := \max_{\|x\|\le\rho'} \|\mathbf{D}A(x)\|, \qquad \lambda_B' := \max_{\|z\|\le\rho'} \|\mathbf{D}B(z)\|, \tag{4.90}$$

and $\lambda_f, \lambda_A, \lambda_h, \lambda_B$ *were defined in (4.29).*

Definition 6. ***(Gradient Mapping)*** *Given* $x,z \in \mathbb{R}^d$ *and* $\gamma,\iota > 0$*, the gradient mappings* $G_{\beta,\gamma}(\cdot,z,y), G_{\beta,\iota}(x,\cdot,y) : \mathbb{R}^d \to \mathbb{R}^d$ *take, respectively,* $x,z \in \mathbb{R}^d$ *to*

$$G_{\beta,\gamma}(x,z,y) = \frac{x-x^+}{\gamma}, \qquad H_{\beta,\iota}(x,z,y) = \frac{z-z^+}{\iota}, \tag{4.91}$$

where $x^+ = P_{\gamma,g}(x - \gamma\nabla_x \mathscr{L}_\beta(x,z,y))$ *and* $z^+ = P_{\iota,l}(z - \iota\nabla_z \mathscr{L}_\beta(x,z,y))$*.*

Lemma 4.11.2 (Descent lemma)**.** *For* $x,z \in \mathbb{R}^d$ *and* $y \in \mathbb{R}^m$*, let* x^+, z^+ *be as in Definition 6 with* $\gamma < 1/\lambda_{\beta,x}$ *and* $\iota < 1/\lambda_{\beta,z}$*. For* $\rho,\rho' \ge 0$*, suppose that*

$$(x,z),(x^+,z),(x,z^+) \in X_{\rho,\rho'}. \tag{4.92}$$

Then it holds that

$$\|G_{\beta,\gamma}(x,z,y)\|^2 \le \frac{2}{\gamma}(\mathscr{L}_\beta(x,z,y) + g(x) - \mathscr{L}_\beta(x^+,z,y) - g(x^+)),$$

$$\|H_{\beta,\iota}(x,z,y)\|^2 \le \frac{2}{\iota}(\mathscr{L}_\beta(x,z,y) + l(x) - \mathscr{L}_\beta(x,z^+,y) - l(x^+)). \tag{4.93}$$

Lemma 4.11.3 (Line search)**.** *Fix* $\theta \in (0,1)$ *and* $\gamma_0,\iota_0 > 0$*. For* $\gamma',\iota' > 0$*, let*

$$x^+_{\gamma'} = P_g(x - \gamma'\nabla_x \mathscr{L}_\beta(x,z,y)), \qquad z^+_{\iota'} = P_l(z - \iota'\nabla_z \mathscr{L}_\beta(x,z,y)), \tag{4.94}$$

and define

$$\gamma := \max\Big\{\gamma' = \gamma_0\theta^i : \mathscr{L}_\beta(x^+_{\gamma'},z,y) \le \mathscr{L}_\beta(x,z,y) + \Big\langle x^+_{\gamma'} - x, \nabla_x \mathscr{L}_\beta(x,z,y)\Big\rangle + \frac{1}{2\gamma'}\|x^+_{\gamma'} - x\|^2\Big\},$$

$$\iota := \max\Big\{\iota' = \iota_0\theta^i : \mathscr{L}_\beta(x, z_{\iota'}^+, y) \\ \leq \mathscr{L}_\beta(x, z, y) + \big\langle z_{\iota'}^+ - z, \nabla_z \mathscr{L}_\beta(x, z, y)\big\rangle + \frac{1}{2\iota'}\|z_{\iota'}^+ - z\|^2\Big\}. \tag{4.95}$$

Then, (4.93) holds and, moreover, we have that

$$\gamma \geq \frac{\theta}{\lambda_{\beta,x}}, \qquad \iota \geq \frac{\theta}{\lambda_{\beta,z}}. \tag{4.96}$$

For the reader's convenience, let us also recall the updates of Algorithm 4 in iteration k as

$$\begin{aligned} x_{k+1} &= P_g(x_k - \gamma_k \nabla_x \mathscr{L}_{\beta_k}(x_k, z_k, y_k)), \\ z_{k+1} &= P_l(z_k - \iota_k \nabla_z \mathscr{L}_{\beta_k}(x_{k+1}, z_k, y_k)), \\ y_{k+1} &= y_k + \sigma_{k+1}(A(x_{k+1}) + B(z_{k+1})). \end{aligned} \tag{4.97}$$

For every $k \in K = [k_0 : k_1]$, recall that the primal step sizes γ_k, ι_k are determined by line search in Lemma 4.11.3. Moreover, the penalty weights and dual step sizes are set as

$$\begin{aligned} \beta_k &= \beta_{k_0}\sqrt{\frac{k\log^2(k+1)}{k_0\log^2(k_0+1)}}, \\ \sigma_k &= \sigma_{k_0}\min\left(\sqrt{\frac{k_0}{k}}, \frac{\|A(x_{k_0}) + B(z_{k_0})\|}{\|A(x_k) + B(z_k)\|}\cdot\frac{k_0\log^2(k_0+1)}{k\log^2(k+1)}\right). \end{aligned} \tag{4.98}$$

For every $k \in K$, let us set

$$G_k = G_{\beta_k,\gamma_k}(x_k, z_k, y_k) = \frac{x_k - x_{k+1}}{\gamma_k},$$

$$H_k = H_{\beta_k,\iota_k}(x_{k+1}, z_k, y_k) = \frac{z_k - z_{k+1}}{\iota_k}, \tag{4.99}$$

for short. The convergence analysis of Algorithm 4 only slightly differs from the one in the proof of Theorem 4.3.2 and we therefore only present the proof sketch, somewhat informally. Similar to the proof of Theorem 4.3.2, two applications of Lemma 4.11.2 yields that

$$\begin{aligned} \frac{\gamma_k\|G_k\|^2}{2} &\leq \mathscr{L}_{\beta_k}(x_k, z_k, y_k) + g(x_k) - \mathscr{L}_{\beta_k}(x_{k+1}, z_k, y_k) - g(x_{k+1}), \\ \frac{\iota_k\|H_k\|^2}{2} &\leq \mathscr{L}_{\beta_k}(x_{k+1}, z_k, y_k) + l(z_k) - \mathscr{L}_{\beta_k}(x_{k+1}, z_{k+1}, y_k) - l(z_{k+1}), \end{aligned} \tag{4.100}$$

for every k. By setting

$$u_k = [\ x_k^\top \quad z_k^\top\ ]^\top \in \mathbb{R}^{2d}, \qquad Q_k = [\ G_k^\top \quad H_k^\top\ ]^\top \in \mathbb{R}^{2d}, \qquad q(u) = f(x) + h(z),$$

$$q(u) = g(x) + l(z), \qquad D(u) = A(x) + B(z), \qquad \kappa_k = \min(\gamma_k, \iota_k), \tag{4.101}$$

for every $k \in K$ and after summing up the two inequalities in (4.100), we reach

$$\frac{\kappa_k \|Q_k\|^2}{2} \leq \mathscr{L}_{\beta_k}(u_k, y_k) + q(u_k) - \mathscr{L}_{\beta_k}(u_{k+1}, y_k) - q(u_{k+1}), \qquad \forall k \in K. \tag{4.102}$$

By following the same steps as in the proof of Theorem 4.3.2, we find that

$$\sum_{k=k_0}^{k_1} \frac{\kappa_k \|Q_k\|^2}{2} \leq \mu + 2 \sum_{k=k_0}^{k_1} \|D(u_k)\|^2, \tag{4.103}$$

where

$$\begin{aligned} \mu := \max\Bigg(\sup_k \Big(& q(u_{k_0}) + q'(u_{k_0}) - q(u_k) - q'(u_k) \\ & + \langle D(u_{k_0}) - D(u_k), y_{k_0} \rangle + \frac{\beta_{k_0}}{2} \|A(x_{k_0}) + B(z_{k_0})\|^2 \Big), 0 \Bigg) < \infty. \end{aligned} \tag{4.104}$$

On the other hand, the x and z updates in (4.97) imply that

$$\begin{aligned} & G_k - \nabla f(x_k) - \mathbf{D} A(x_k)^\top y_k \\ & \qquad - \beta_k \mathbf{D} A(x_k)^\top (A(x_k) + B(z_k)) \in \frac{\partial g(x_{k+1})}{\gamma_k} \subseteq \frac{\partial g(x_{k+1})}{\kappa_k}, \end{aligned} \tag{4.105}$$

$$\begin{aligned} & H_k - \nabla h(z_k) - \mathbf{D} B(z_k)^\top y_k \\ & \qquad - \beta_k \mathbf{D} B(z_k)^\top (A(x_{k+1}) + B(z_k)) \in \frac{\partial l(z_{k+1})}{\iota_k} \subseteq \frac{\partial l(x_{k+1})}{\kappa_k}, \end{aligned} \tag{4.106}$$

which can be more compactly written as

$$\begin{aligned} & Q_k - \nabla q(u_k) - \mathbf{D} D(u_k)^\top y_k - \beta_k \mathbf{D} D(u_k)^\top D(u_k) \\ & \qquad + \beta_k \begin{bmatrix} 0 \\ \mathbf{D} B(z_k)^\top (A(x_k) - A(x_{k+1})) \end{bmatrix} \in \frac{\partial q(u_{k+1})}{\kappa_k}. \end{aligned} \tag{4.107}$$

Note that (4.107) is similar to (4.122), except for its last term, which satisfies

$$\begin{aligned} & \beta_k \left\| \begin{bmatrix} 0 \\ \mathbf{D} B(z_k)^\top (A(x_k) - A(x_{k+1})) \end{bmatrix} \right\| \\ & = \beta_k \|\mathbf{D} B(z_k)^\top (A(x_k) - A(x_{k+1}))\| \\ & \leq \beta_k \lambda'_A \lambda'_B \|x_k - x_{k+1}\| \qquad \text{(see (4.90))} \\ & = \beta_k \gamma_k \lambda'_A \lambda'_B \|G_k\| \qquad \text{(see (4.91))} \\ & \leq \beta_k \gamma_k \lambda'_A \lambda'_B \|Q_k\|. \qquad \text{(see (4.101))} \end{aligned} \tag{4.108}$$

From this point, the rest of the proof steps of Lemma 4.3.1 and Theorem 4.3.2 can be applied directly to complete the proof of Theorem 4.4.1.

4.12 Proof of Lemma 4.1.1

If A_i denotes the i^{th} component of the map $A:\mathbb{R}^d \to \mathbb{R}^m$, note that

$$\mathscr{L}_\beta(x,y) = f(x) + \sum_{i=1}^m y_i A_i(x) + \frac{\beta}{2}\sum_{i=1}^m (A_i(x))^2, \tag{4.109}$$

which implies that

$$\begin{aligned}\nabla_x \mathscr{L}_\beta(x,y) &= \nabla f(x) + \sum_{i=1}^m y_i \nabla A_i(x) + \frac{\beta}{2}\sum_{i=1}^m A_i(x)\nabla A_i(x)\\ &= \nabla f(x) + \mathbf{D}A(x)^\top y + \beta \mathbf{D}A(x)^\top A(x),\end{aligned} \tag{4.110}$$

where $DA(x)$ is the Jacobian of A at x. By taking another derivative with respect to x, we reach

$$\nabla^2_{xx}\mathscr{L}_\beta(x,y) = \nabla^2_{xx} f(x) + \sum_{i=1}^m \big(y_i + \beta A_i(x)\big)\nabla^2_{xx} A_i(x) + \beta \sum_{i=1}^m \nabla_x A_i(x)\nabla_x A_i(x)^\top. \tag{4.111}$$

It follows that

$$\begin{aligned}\|\nabla^2_{xx}\mathscr{L}_\beta(x,y)\| &\le \|\nabla^2_{xx} f(x)\| + \max_i \|\nabla^2_{xx} A_i(x)\|\big(\|y\|_1 + \beta\|A(x)\|_1\big) + \beta\sum_{i=1}^m \|\nabla_x A_i(x)\|^2\\ &\le \lambda_h + \sqrt{m}\lambda_A\big(\|y\| + \beta\|A(x)\|\big) + \beta\|\mathbf{D}A(x)\|_F^2.\end{aligned} \tag{4.112}$$

For every x such that $\|A(x)\| \le \rho$ and $\|x\| \le \rho'$, we conclude that

$$\|\nabla^2_{xx}\mathscr{L}_\beta(x,y)\| \le \lambda_f + \sqrt{m}\lambda_A\big(\|y\| + \beta\rho\big) + \beta \max_{\|x\|\le\rho'} \|\mathbf{D}A(x)\|_F^2, \tag{4.113}$$

which completes the proof of Lemma 4.1.1.

4.13 Proof of Lemma 4.1.2

Throughout, let

$$G = G_{\beta,\gamma}(x,y) = \frac{x - x^+}{\gamma}, \tag{4.114}$$

for short. Suppose that $\|A(x)\| \le \rho$, $\|x\| \le \rho'$, and similarly $\|A(x^+)\| \le \rho$, $\|x^+\| \le \rho'$. An application of Lemma 4.1.1 yields that

$$\begin{aligned}\mathscr{L}_\beta(x^+, y) + g(x^+) &\le \mathscr{L}_\beta(x, y) + \langle x^+ - x, \nabla_x \mathscr{L}_\beta(x, y)\rangle + \frac{\lambda_\beta}{2}\|x^+ - x\|^2 + g(x^+) \\ &= \mathscr{L}_\beta(x, y) - \gamma\langle G, \nabla_x \mathscr{L}_\beta(x, y)\rangle + \frac{\gamma^2\lambda_\beta}{2}\|G\|^2 + g(x^+) \end{aligned} \tag{4.115}$$

Since $x^+ = P_g(x - \gamma\nabla_x \mathscr{L}_\beta(x, y))$, we also have that

$$G - \nabla_x \mathscr{L}_\beta(x, y) = \xi \in \partial g(x^+). \tag{4.116}$$

By combining (4.115,4.116), we find that

$$\begin{aligned}\mathscr{L}_\beta(x^+, y) + g(x^+) &\le \mathscr{L}_\beta(x, y) - \gamma\|G\|^2 + \gamma\langle G, \xi\rangle + \frac{\gamma^2\lambda_\beta}{2}\|G\|^2 + g(x^+) \\ &= \mathscr{L}_\beta(x, y) - \gamma\|G\|^2 + \langle x - x^+, \xi\rangle + \frac{\gamma^2\lambda_\beta}{2}\|G\|^2 + g(x^+) \\ &\le \mathscr{L}_\beta(x, y) + g(x) - \gamma\left(1 - \frac{\gamma\lambda_\beta}{2}\right)\|G\|^2, \end{aligned} \tag{4.117}$$

where the last line above uses the convexity of g. Recalling that $\gamma \le 1/\lambda_\beta$ completes the proof of Lemma 4.1.2.

4.14 Proof of Lemma 4.1.3

By optimality of x_γ^+ , we note that

$$x_\gamma^+ - x + \gamma\nabla_x \mathscr{L}_\beta(x, y) = -\gamma\xi \in -\gamma\partial g(x_\gamma^+). \tag{4.118}$$

By definition in (4.12), γ also satisfies

$$\begin{aligned}&\mathscr{L}_\beta(x_\gamma^+, y) + g(x_\gamma^+) \\ &\le \mathscr{L}_\beta(x, y) + \left\langle x_\gamma^+ - x, \nabla_x \mathscr{L}_\beta(x, y)\right\rangle + \frac{1}{2\gamma}\|x_\gamma^+ - x\|^2 + g(x_\gamma^+) \\ &= \mathscr{L}_\beta(x, y) + \left\langle x - x_\gamma^+, \xi\right\rangle - \frac{1}{2\gamma}\|x_\gamma^+ - x\|^2 + g(x_\gamma^+) \\ &\le \mathscr{L}_\beta(x, y) - \frac{1}{2\gamma}\|x_\gamma^+ - x\|^2 + g(x) \qquad \text{(convexity of } g\text{)} \\ &= \mathscr{L}_\beta(x, y) - \frac{\gamma}{2}\|G_{\beta,\gamma}(x, y)\|^2 + g(x), \qquad \text{(see Definition 1)} \end{aligned} \tag{4.119}$$

which completes the proof of Lemma 4.1.3 since $\gamma \ge \frac{\theta}{\lambda_\beta}$follows directly from (4.12).

4.15 Proof of Lemma 4.3.1

By assumption, we have that

$$\max_{k\in K}\|A(x_k)\| \le \rho, \qquad \max_{k\in K}\|x_k\| \le \rho'. \tag{4.120}$$

From the primal update in (4.50) and the definition of the proximal operator P_g [PB$^+$14], it follows that

$$x_{k+1} - x_k + \gamma_k \nabla f(x_k) + \gamma_k \mathbf{D}A(x_k)^\top (y_k + \beta_k A(x_k)) \in -\partial g(x_{k+1}), \tag{4.121}$$

where $\partial g(x_{k+1})$ is the subdifferential of g at x_{k+1}. After recalling the definition of gradient mapping in (4.52), the above inclusion can be written as

$$-\frac{G_k}{\beta_k} + \frac{\nabla f(x_k)}{\beta_k} + \frac{\mathbf{D}A(x_k)^\top y_k}{\beta_k} + \mathbf{D}A(x_k)^\top A(x_k) \in -\frac{\partial g(x_{k+1})}{\beta_k \gamma_k}. \tag{4.122}$$

Let $\mathrm{cone}(\partial g(x))^*$ denote the polar of

$$\mathrm{cone}(\partial g(x))) = \bigcup_{\alpha\ge 0} \alpha \cdot \partial g(x) \subseteq \mathbb{R}^d. \tag{4.123}$$

By projecting both sides (4.122) onto $\mathrm{cone}(\partial g(x_{k+1}))^*$, we find that

$$\begin{aligned} &P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(-\frac{G_k}{\beta_k} + \frac{\nabla f(x_k)}{\beta_k} + \frac{\mathbf{D}A(x_k)^\top y_k}{\beta_k} + \mathbf{D}A(x_k)^\top A(x_k)\right) \\ &\in P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(-\frac{\partial g(x_{k+1})}{\beta_k \gamma_k}\right) = \{0\}, \end{aligned} \tag{4.124}$$

where the equality above follows from the duality of $\mathrm{cone}(\partial g(x_{k+1}))^*$ and $\mathrm{cone}(\partial g(x_{k+1}))$. Recall also that the subspace $S \subseteq \mathbb{R}^d$ by assumption satisfies

$$S \supseteq \bigcup_{k\in K} P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(\mathbf{D}A(x_{k+1})^\top A(x_{k+1})\right), \tag{4.125}$$

and project both sides of (4.124) onto S to reach

$$P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(-\frac{G_k}{\beta_k} + \frac{\nabla f(x_k)}{\beta_k} + \frac{\mathbf{D}A(x_k)^\top y_k}{\beta_k} + \mathbf{D}A(x_k)^\top A(x_k)\right) = 0. \tag{4.126}$$

When K is a convex cone, recall the property that

$$P_K(a+b) \le P_K(a) + P_K(b), \tag{4.127}$$

for a cone K and vectors a, b. Using this property, then by taking the norm and finally applying the triangle inequality above, we argue that

$$\begin{aligned}
&\left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(\mathbf{D}A(x_k)^\top A(x_k)) \right\| \\
&\leq \left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}\left(-\frac{G_k}{\beta_k} + \frac{\nabla f(x_k)}{\beta_k} + \frac{\mathbf{D}A(x_k)^\top y_k}{\beta_k} \right) \right\| \qquad \text{(see (4.126))}. \qquad (4.128)
\end{aligned}$$

Because proximal map is non-expansive and $P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(0) = 0$, we may upper bound the last line above as

$$\begin{aligned}
&\left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(\mathbf{D}A(x_k)^\top A(x_k)) \right\| \\
&\leq \left\| -\frac{G_k}{\beta_k} + \frac{\nabla f(x_k)}{\beta_k} + \frac{\mathbf{D}A(x_k)^\top y_k}{\beta_k} \right\| \\
&\leq \frac{1}{\beta_k}\left(\|G_k\| + \|\nabla f(x_k)\| + \|\mathbf{D}A(x_k)^\top y_k\| \right). \qquad \text{(triangle inequality)} \\
&\leq \frac{1}{\beta_k}\left(\|G_k\| + \lambda_f' + \lambda_A' \|y_k\| \right), \qquad (4.129)
\end{aligned}$$

where

$$\lambda_f' := \max_{\|x\| \leq \rho'} \|\nabla f(x)\|, \qquad \lambda_A' := \max_{\|x\| \leq \rho'} \|\mathbf{D}A(x)\|, \qquad (4.130)$$

are the local Lipschitz constant of f and A. To lower bound the first line of (4.129), we invoke the restricted injectivity in Section 4.2. Indeed, recalling (4.35) and the first bound in (4.120), for every $k \in K$, we write that

$$\begin{aligned}
&\left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(\mathbf{D}A(x_k)^\top A(x_k)) \right\| \\
&\geq \left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(\mathbf{D}A(x_{k+1})^\top A(x_k)) \right\| - \left\| (\mathbf{D}A(x_{k+1}) - \mathbf{D}A(x_k))^\top A(x_k) \right\| \\
&\geq \nu(g, A, S, \rho, \rho') \|A(x_k)\| - \|\mathbf{D}A(x_{k+1}) - \mathbf{D}A(x_k)\| \|A(x_k)\|, \qquad \text{(see (4.35))} \qquad (4.131)
\end{aligned}$$

where the second line above again uses the non-expansiveness of P_S and $P_{\mathrm{cone}(\partial g(x_{k+1}))^*}$. The remaining term in (4.131) is bounded as

$$\|\mathbf{D}A(x_{k+1}) - \mathbf{D}A(x_k)\| \leq \lambda_A \|x_{k+1} - x_k\| = \lambda_A \gamma_k \|G_k\|. \qquad \text{(see (4.2,4.120))} \qquad (4.132)$$

Assuming that

$$\nu(g, A, S, \rho, \rho') \geq 2\lambda_A \max_{k \in K} \gamma_k \|G_k\|, \qquad (4.133)$$

allows us to simplify the last line of (4.131) as

$$\left\| P_S P_{\mathrm{cone}(\partial g(x_{k+1}))^*}(\mathbf{D}A(x_k)^\top A(x_k)) \right\| \geq \frac{\nu(g, A, S, \rho, \rho')}{2} \|A(x_k)\|, \qquad (4.134)$$

which, after substituting in (4.129), yields that

$$\|A(x_k)\| \leq \frac{2}{\beta_k \nu(g, A, S, \rho, \rho')} \left(\|G_k\| + \lambda'_f + \lambda'_A \|y_k\| \right), \tag{4.135}$$

and completes the proof of Lemma 4.3.1.

4.16 Bibliographic notes

Theorem 4.3.2 and 4.4.1 are joint between Armin Eftekhari and the author of this thesis.

5 Conclusions and future directions

In this chapter, we give an overview of our findings and provide possible future directions that are left open.

In Chapter 2, we proposed and analyzed an inexact augmented Lagrangian method for solving (1.1). We prove convergence to the first and second order stationary points of the augmented Lagrangian function, with explicit complexity estimates. Even though the relation of stationary points and global optima is not well-understood in the literature, we find out that the algorithm has fast convergence behavior to either global minima or local minima in a wide variety of numerical experiments. We found out our complexity results for first order stationary point to be suboptimal.

In Chapter 3, we directed our attention to a more specialized algorithm aimed at solving semidefinite programming (SDP) problems by factorizing the decision variable. The resulting optimization problems are inherently non-convex and nonlinear. We obtain the rate of convergence with an augmented Lagrangian method which combines aspects of both linearized and inexact augmented Lagrangian methods.

In Chapter 4, we proposed algorithms with convergence complexity results to reach to an $\epsilon-$ first order stationarity point of (4.1) and (4.20) with a simple, easy-to-implement linearized augmented Lagrangian algorithm and its alternating direction method-of-multipliers variant subject to a verifiable geometric regularity condition. While the condition in its form is difficult to interpret, we believe that it acts as a constraint qualification condition helping to derive theoretical guarantees. Fortunately, we can verify this condition for a variety of contemporary problems in machine learning including clustering and Max-Cut. We also provide numerical evidence on an interesting causal learning problem without verifying the condition. Finally, we find out that the LAL algorithm and the ADMM variant require very little tuning, which should not be underestimated for large-scale problems, while exhibiting comparable performance against the baselines that feature more tuning parameters.

We believe that the regularity condition we proposed in this thesis has opened doors for developing

algorithms for many interesting problems in the community ranging from multi-class Neyman-Pearson classification [Lu22] to training logical neural networks [LKA+21]. Improving the rate of convergence for the linearized AL is left for future work. Extending the results in this work to the setting where both the objective function and the constraints are stochastic is a challenging research direction which would give much more opportunities to tackle various other interesting problems such as streaming eigenvalue problem and reinforcement learning with large state action pairs [Lu22]. Step size is a tuning parameter for the proposed algorithms. Whether one can adaptively choice the step-size is also an interesting research direction.

Bibliography

[ABKK17] Naman Agarwal, Afonso S Bandeira, Konstantinos Koiliaris, and Alexandra Kolla. Multisection in the stochastic block model using semidefinite programming. In *Compressed Sensing and its Applications: Second International MATHEON Conference 2015*, pages 125–162. Springer, 2017. [Page 1.]

[BAC16] Nicolas Boumal, P-A Absil, and Coralia Cartis. Global rates of convergence for nonconvex optimization on manifolds. *arXiv preprint arXiv:1605.08101*, 2016. [Pages 15, 29, 39, 40, 46, and 47.]

[Bar95] Alexander I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13(2):189–202, 1995. [Pages 8, 15, 32, and 36.]

[BBJN18] Srinadh Bhojanapalli, Nicolas Boumal, Prateek Jain, and Praneeth Netrapalli. Smoothed analysis for low-rank solutions to semidefinite programs in quadratic penalty form. *arXiv preprint arXiv:1803.00186*, 2018. [Pages 20, 36, and 40.]

[BBVP04] S. Boyd, S.P. Boyd, L. Vandenberghe, and Cambridge University Press. *Convex Optimization*. Berichte uber verteilte messysteme. Cambridge University Press, 2004. [Page 54.]

[Ber82] Dimitri P Bertsekas. Constrained optimization and lagrange multiplier methods. *Computer Science and Applied Mathematics, Boston: Academic Press, 1982*, 1982. [Pages 3 and 9.]

[Ber99] Dimitri P Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999. [Page 39.]

[Ber14] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014. [Pages 14, 36, 38, 39, 53, and 54.]

[BGM+16] Ernesto G Birgin, JL Gardenghi, José Mario Martinez, SA Santos, and Ph L Toint. Evaluation complexity for nonlinear constrained optimization using unscaled kkt conditions and high-order models. *SIAM Journal on Optimization*, 26(2):951–967, 2016. [Pages 2, 13, 14, 39, and 40.]

[BIL+16] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya V. Nori, and Antonio Criminisi. Measuring neural net robustness with constraints. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 2621–2629, Red Hook, NY, USA, 2016. Curran Associates Inc. [Page 52.]

[BKS16] Srinadh Bhojanapalli, Anastasios Kyrillidis, and Sujay Sanghavi. Dropping convexity for faster semi-definite optimization. In *Conference on Learning Theory*, pages 530–582. PMLR, 2016. [Pages 15, 36, and 39.]

[BM03a] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Math. Program.*, 95(2, Ser. B):329–357, 2003. [Page 36.]

[BM03b] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003. [Pages 1, 8, 14, 36, 37, 39, 46, and 49.]

[BM05] Samuel Burer and Renato DC Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005. [Pages 8, 14, 36, 37, 39, 46, and 49.]

[BM14] Ernesto G Birgin and Josâ Mario Mart_nez. *Practical augmented Lagrangian methods for constrained optimization*, volume 10. SIAM, 2014. [Page 3.]

[BMAS14] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research*, 15(1):1455–1459, 2014. [Pages 15, 29, 39, 40, 46, and 47.]

[BN18] Radu Ioan Bot and Dang-Khoa Nguyen. The proximal alternating direction method of multipliers in the nonconvex setting: convergence analysis and rates. *arXiv preprint arXiv:1801.01994*, 2018. [Pages 52 and 62.]

[BNPS17] Jérôme Bolte, Trong Phong Nguyen, Juan Peypouquet, and Bruce W Suter. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming*, 165(2):471–507, 2017. [Page 14.]

[Bou15] N. Boumal. A Riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints. Available at http://arXiv.org/abs/1506.00575, June 2015. [Page 36.]

[BPC+11] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011. [Page 58.]

[BST14] Jérôme Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, 2014. [Page 54.]

[BST18] Jerome Bolte, Shoham Sabach, and Marc Teboulle. Nonconvex lagrangian-based optimization: monitoring schemes and global convergence. *Mathematics of Operations Research*, 2018. [Pages 2, 12, 14, 53, and 65.]

[BVB16] Nicolas Boumal, Vlad Voroninski, and Afonso Bandeira. The non-convex burer-monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems*, pages 2757–2765, 2016. [Pages 1, 8, 15, 27, 47, and 48.]

[CDZ15] Nikolaos Chatzipanagiotis, Darinka Dentcheva, and Michael M Zavlanos. An augmented lagrangian method for distributed optimization. *Mathematical Programming*, 152:405–434, 2015. [Page 3.]

[CGT11] Coralia Cartis, Nicholas IM Gould, and Philippe L Toint. On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming. *SIAM Journal on Optimization*, 21(4):1721–1739, 2011. [Pages 2 and 14.]

[CGT12] Coralia Cartis, Nicholas IM Gould, and Ph L Toint. Complexity bounds for second-order optimality in unconstrained optimization. *Journal of Complexity*, 28(1):93–108, 2012. [Pages 12, 13, 15, 19, and 20.]

[CGT18] Coralia Cartis, Nicholas IM Gould, and Ph L Toint. Optimality of orders one to three and beyond: characterization and evaluation complexity in constrained nonconvex optimization. *Journal of Complexity*, 2018. [Pages 2, 13, and 14.]

[Cif21] Diego Cifuentes. On the burer–monteiro method for general semidefinite programs. *Optimization Letters*, pages 1–11, 2021. [Page 15.]

[CJG+19] Andrew Cotter, Heinrich Jiang, Maya R Gupta, Serena Wang, Taman Narayan, Seungil You, and Karthik Sridharan. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *J. Mach. Learn. Res.*, 20(172):1–59, 2019. [Page 1.]

[CKS15] Kunal N Chaudhury, Yuehaw Khoo, and Amit Singer. Global registration of multiple point clouds using semidefinite programming. *SIAM Journal on Optimization*, 25(1):468–501, 2015. [Page 35.]

[CLS15] E. J. Candès, X. Li, and M. Soltanolkoltabi. Phase retrieval via Wirtinger Flow: Theory and algorithms. *IEEE Trans. Inform. Theory*, 61(4):1985–2007, 2015. [Page 36.]

[CMV18] Christian Clason, Stanislav Mazurenko, and Tuomo Valkonen. Acceleration and global convergence of a first-order primal–dual method for nonconvex problems. *arXiv preprint arXiv:1802.03347*, 2018. [Page 14.]

[CMV19] Christian Clason, Stanislav Mazurenko, and Tuomo Valkonen. Acceleration and global convergence of a first-order primal-dual method for nonconvex problems. *SIAM Journal on Optimization*, 29(1):933–963, 2019. [Page 40.]

[CO19] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32, 2019. [Page 1.]

[CP11] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011. [Pages 14, 25, and 40.]

[Dai02] Yu-Hong Dai. Convergence properties of the bfgs algoritm. *SIAM Journal on Optimization*, 13(3):693–701, 2002. [Page 15.]

[DH11] Timothy A Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1, 2011. [Page 47.]

[Doz16] Timothy Dozat. Incorporating nesterov momentum into adam. 2016. [Page 1.]

[FBFBV12] Fabián Flores-Bazán, Fernando Flores-Bazán, and Cristián Vera. A complete characterization of strong duality in nonconvex optimization with a single constraint. *Journal of Global Optimization*, 53(2):185–201, 2012. [Page 53.]

[FKS18] José FS Bravo Ferreira, Yuehaw Khoo, and Amit Singer. Semidefinite programming approach for the quadratic assignment problem with a sparse graph. *Computational Optimization and Applications*, 69(3):677–712, 2018. [Pages 32 and 33.]

[Fle13] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013. [Page 17.]

[FMP20] Mahyar Fazlyab, Manfred Morari, and George J Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 67(1):1–15, 2020. [Page 1.]

[FS12] Damian Fernandez and Mikhail V Solodov. Local convergence of exact and inexact augmented lagrangian methods under the second-order sufficient optimality condition. *SIAM Journal on Optimization*, 22(2):384–407, 2012. [Page 14.]

[GJN+16] Rong Ge, Chi Jin, Praneeth Netrapalli, Aaron Sidford, et al. Efficient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis. In *International Conference on Machine Learning*, pages 2741–2750, 2016. [Pages 27, 29, and 48.]

[GL16] Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016. [Pages 12, 13, and 18.]

[GLZ16] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305, 2016. [Pages 4 and 42.]

[GM75] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 9(R2):41–76, 1975. [Page 58.]

[GM76] Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2:17–40, 12 1976. [Page 58.]

[GPM+14] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. *ArXiv e-prints*, June 2014. [Pages 30 and 62.]

[Gro96] Alexander Grothendieck. Résumé de la théorie métrique des produits tensoriels topologiques. *Resenhas do Instituto de Matemática e Estatistica da Universidade de Sao Paulo*, 2(4):401–481, 1996. [Page 36.]

[GW95] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995. [Page 1.]

[Hes69] Magnus R Hestenes. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320, 1969. [Pages 3, 14, 36, 38, 39, and 52.]

[HH19] Davood Hajinezhad and Mingyi Hong. Perturbed proximal primal–dual algorithm for nonconvex nonsmooth optimization. *Mathematical Programming*, 176(1-2):207–245, 2019. [Page 2.]

[HL17] Mingyi Hong and Zhi-Quan Luo. On the linear convergence of the alternating direction method of multipliers. *Mathematical Programming*, 162(1):165–199, 2017. [Page 38.]

[HLR16] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016. [Pages 2, 52, and 62.]

[IJA+17] Andrew Ilyas, Ajil Jalal, Eirini Asteri, Constantinos Daskalakis, and Alexandros G. Dimakis. The Robust Manifold Defense: Adversarial Training using Generative Models. *arXiv e-prints*, page arXiv:1712.09196, December 2017. [Page 30.]

[Jag13] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pages 427–435, 2013. [Page 15.]

[JGN+17] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pages 1724–1732. PMLR, 2017. [Page 36.]

[JNS13] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013. [Page 36.]

[KB14] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv e-prints*, page arXiv:1412.6980, December 2014. [Pages 1 and 30.]

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. [Page 71.]

[KMM19] Weiwei Kong, Jefferson G Melo, and Renato DC Monteiro. Complexity of a quadratic penalty accelerated inexact proximal point method for solving linearly constrained nonconvex composite programs. *SIAM Journal on Optimization*, 29(4):2566–2593, 2019. [Page 61.]

[KN11] Subhash Khot and Assaf Naor. Grothendieck-type inequalities in combinatorial optimization. *arXiv preprint arXiv:1108.2464*, 2011. [Page 7.]

[KN12] Subhash Khot and Assaf Naor. Grothendieck-type inequalities in combinatorial optimization. *Communications on Pure and Applied Mathematics*, 65(7):992–1035, 2012. [Pages 35 and 51.]

[KNS16] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016. [Pages 9, 12, 53, and 65.]

[KSP07] Brian Kulis, Arun C Surendran, and John C Platt. Fast low-rank semidefinite programming for embedding and clustering. In *Artificial Intelligence and Statistics*, pages 235–242, 2007. [Pages 8, 16, and 17.]

[LAL+21] Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, and Mykel J. Kochenderfer. Algorithms for verifying deep neural networks. *Foundations and Trends® in Optimization*, 4(3-4):244–404, 2021. [Page 52.]

[LC+19] Fabian Latorre, Volkan Cevher, et al. Fast and provable admm for learning with generative priors. *Advances in Neural Information Processing Systems*, 32, 2019. [Page 58.]

[LCL+21] Zichong Li, Pin-Yu Chen, Sijia Liu, Songtao Lu, and Yangyang Xu. Rate-improved inexact augmented lagrangian method for constrained nonconvex optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 2170–2178. PMLR, 2021. [Pages 52, 55, 61, 63, and 64.]

[LFJ11] Fu Lin, Makan Fardad, and Mihailo R Jovanovic. Augmented lagrangian approach to design of structured optimal state feedback gains. *IEEE Transactions on Automatic Control*, 56(12):2923–2929, 2011. [Page 3.]

[LJJ20a] Tianyi Lin, Chi Jin, and Michael Jordan. On gradient descent ascent for nonconvex-concave minimax problems. In *International Conference on Machine Learning*, pages 6083–6093. PMLR, 2020. [Page 62.]

[LJJ20b] Tianyi Lin, Chi Jin, and Michael I Jordan. Near-optimal algorithms for minimax optimization. In *Conference on Learning Theory*, pages 2738–2779. PMLR, 2020. [Page 62.]

[LKA+21] Songtao Lu, Naweed Khan, Ismail Yunus Akhalwaya, Ryan Riegel, Lior Horesh, and Alexander Gray. Training logical neural networks by primal–dual methods for neuro-symbolic reasoning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5559–5563. IEEE, 2021. [Page 92.]

[LM16] Guanghui Lan and Renato DC Monteiro. Iteration-complexity of first-order augmented lagrangian methods for convex programming. *Mathematical Programming*, 155(1-2):511–547, 2016. [Pages 8, 14, and 39.]

[LMX19] Qihang Lin, Runchao Ma, and Yangyang Xu. Inexact proximal-point penalty methods for constrained non-convex optimization. *arXiv preprint arXiv:1908.11518*, 2019. [Page 61.]

[Lov03] László Lovász. Semidefinite programs and combinatorial optimization. In *Recent advances in algorithms and combinatorics*, pages 137–194. Springer, 2003. [Pages 7, 35, and 51.]

[LSG17] Qinghua Liu, Xinyue Shen, and Yuantao Gu. Linearized admm for non-convex non-smooth optimization with convergence analysis. *arXiv preprint arXiv:1705.02502*, 2017. [Page 39.]

[LSG19] Qinghua Liu, Xinyue Shen, and Yuantao Gu. Linearized admm for nonconvex nonsmooth optimization with convergence analysis. *IEEE Access*, 2019. [Pages 52 and 62.]

[LTHC20] Songtao Lu, Ioannis Tsaknakis, Mingyi Hong, and Yongxin Chen. Hybrid block successive approximation for one-sided non-convex min-max problems: algorithms and applications. *IEEE Transactions on Signal Processing*, 68:3676–3691, 2020. [Page 51.]

[Lu22] Songtao Lu. A single-loop gradient descent and perturbed ascent algorithm for nonconvex functional constrained optimization. In *International Conference on Machine Learning*, pages 14315–14357. PMLR, 2022. [Pages 61 and 92.]

[LX20] Zichong Li and Yangyang Xu. Augmented lagrangian based first-order methods for convex and nonconvex programs: nonergodic convergence and iteration complexity. *arXiv preprint arXiv:2003.08880*, 2020. [Page 61.]

[Mas04] Walter F Mascarenhas. The bfgs method with exact line searches fails for non-convex objective functions. *Mathematical Programming*, 99(1):49–61, 2004. [Page 15.]

[MF67] Olvi L Mangasarian and Stan Fromovitz. The fritz john necessary optimality conditions in the presence of equality and inequality constraints. *Journal of Mathematical Analysis and applications*, 17(1):37–47, 1967. [Page 54.]

[MLY20] Runchao Ma, Qihang Lin, and Tianbao Yang. Quadratically regularized subgradient methods for weakly convex optimization with weakly convex constraints. In *International Conference on Machine Learning*, pages 6554–6564. PMLR, 2020. [Page 61.]

[MNS15] Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for the planted bisection model. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 69–75, 2015. [Pages 7, 35, and 51.]

[Mor62] Jean Jacques Moreau. D'ecomposition orthgonale d"un espace hilbertien selon deux cones mutuellement polaires. In *D'ecomposition orthgonale d"un espace hilbertien selon deux cones mutuellement polaires*, 1962. [Pages 55 and 66.]

[MVW17] Dustin G Mixon, Soledad Villar, and Rachel Ward. Clustering subgaussian mixtures by semidefinite programming. *Information and Inference: A Journal of the IMA*, 6(4):389–415, 2017. [Pages 17 and 49.]

[Nes83] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547, 1983. [Page 18.]

[Nes09] Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009. [Page 15.]

[NLR18] Maher Nouiehed, Jason D Lee, and Meisam Razaviyayn. Convergence to second-order stationarity for constrained non-convex optimization. *arXiv preprint arXiv:1810.02024*, 2018. [Pages 19 and 20.]

[NNTD14] Valentin Nedelcu, Ion Necoara, and Quoc Tran-Dinh. Computational complexity of inexact gradient augmented lagrangian methods: application to constrained mpc. *SIAM Journal on Control and Optimization*, 52(5):3109–3134, 2014. [Pages 8, 14, 39, and 52.]

[NW06] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006. [Page 8.]

[OJV11] Guillaume Obozinski, Laurent Jacob, and Jean-Philippe Vert. Group lasso with overlaps: the latent group lasso approach. *arXiv preprint arXiv:1110.0413*, 2011. [Page 26.]

[Pat98] Gábor Pataki. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998. [Pages 8, 15, 32, and 36.]

[PB+14] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014. [Pages 7, 8, and 86.]

[PKB+16] Dohyung Park, Anastasios Kyrillidis, Srinadh Bhojanapalli, Constantine Caramanis, and Sujay Sanghavi. Provable burer-monteiro factorization for a class of norm-constrained matrix problems. *arXiv preprint arXiv:1606.01316*, 2016. [Pages 15 and 40.]

[Pow69] Michael JD Powell. A method for nonlinear constraints in minimization problems. *Optimization*, pages 283–298, 1969. [Pages 3 and 14.]

[Pow78] Michael JD Powell. A fast algorithm for nonlinearly constrained optimization calculations. In *Numerical analysis*, pages 144–157. Springer, 1978. [Pages 36, 38, and 39.]

[PW07] J. Peng and Y. Wei. Approximating K–means–type clustering via semidefinite programming. *SIAM J. Optim.*, 18(1):186–205, 2007. [Pages 16, 62, 63, and 67.]

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254, 2008. [Pages 8 and 36.]

[RKK19] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019. [Page 1.]

[RMC15] A. Radford, L. Metz, and S. Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *ArXiv e-prints*, November 2015. [Page 30.]

[Roc70] R Tyrrell Rockafellar. *Convex analysis*, volume 28. Princeton university press, 1970. [Page 13.]

[Roc93] R Tyrrell Rockafellar. Lagrange multipliers and optimality. *SIAM review*, 35(2):183–238, 1993. [Pages 12 and 53.]

[Roc15] R.T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 2015. [Page 66.]

[SAL+19] Mehmet Fatih Sahin, Ahmet Alacaoglu, Fabian Latorre, Volkan Cevher, et al. An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints. In *Advances in Neural Information Processing Systems*, pages 13943–13955, 2019. [Pages 52, 55, 61, 63, 64, 65, 67, and 69.]

[SGSH00] Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000. [Page 71.]

[Sin11] Amit Singer. Angular synchronization by eigenvectors and semidefinite programming. *Applied and computational harmonic analysis*, 30(1):20–36, 2011. [Pages 7, 35, and 51.]

[SKC18] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018. [Page 30.]

[SPP+05] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005. [Page 71.]

[SS11] Amit Singer and Yoel Shkolnisky. Three-dimensional structure determination from common lines in cryo-em by eigenvectors and semidefinite programming. *SIAM journal on imaging sciences*, 4(2):543–572, 2011. [Pages 7, 35, and 51.]

[SSGB07] Le Song, Alex Smola, Arthur Gretton, and Karsten M Borgwardt. A dependence maximization view of clustering. In *Proceedings of the 24th international conference on Machine learning*, pages 815–822. ACM, 2007. [Pages 7, 35, and 51.]

[TDAFC18] Quoc Tran-Dinh, Ahmet Alacaoglu, Olivier Fercoq, and Volkan Cevher. An adaptive primal-dual framework for nonsmooth convex minimization. *arXiv preprint arXiv:1808.04648*, 2018. [Pages 14 and 25.]

[TDFC18] Quoc Tran-Dinh, Olivier Fercoq, and Volkan Cevher. A smooth primal-dual optimization framework for nonsmooth composite convex minimization. *SIAM Journal on Optimization*, 28(1):96–134, 2018. [Pages 20, 25, 46, and 52.]

[TH12] Tijmen Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *COURSERA Neural Networks Mach. Learn*, 17, 2012. [Page 1.]

[TSC18] Mariano Tepper, Anirvan M Sengupta, and Dmitri Chklovskii. Clustering is semidefinitely not that hard: Nonnegative sdp for manifold disentangling. *Journal of Machine Learning Research*, 19(82), 2018. [Page 17.]

[WW18] Irène Waldspurger and Alden Waters. Rank optimality for the burer-monteiro factorization. *arXiv preprint arXiv:1812.03046*, 2018. [Page 8.]

[WYZ15] Yu Wang, Wotao Yin, and Jinshan Zeng. Global convergence of admm in nonconvex nonsmooth optimization. *arXiv preprint arXiv:1511.06324*, 2015. [Page 39.]

[WZ21] Junxiang Wang and Liang Zhao. Nonconvex generalization of alternating direction method of multipliers for nonlinear equality constrained problems. *Results in Control and Optimization*, 2:100009, 2021. [Page 62.]

[XRV17] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. [Page 17.]

[Xu17a] Yangyang Xu. Accelerated first-order primal-dual proximal methods for linearly constrained composite convex programming. *SIAM Journal on Optimization*, 27(3):1459–1484, 2017. [Page 64.]

[Xu17b] Yangyang Xu. Iteration complexity of inexact augmented lagrangian methods for constrained convex programming. *arXiv preprint arXiv:1711.05812v2*, 2017. [Pages 8 and 14.]

[Xu21] Yangyang Xu. Iteration complexity of inexact augmented lagrangian methods for constrained convex programming. *Mathematical Programming*, 185:199–244, 2021. [Page 2.]

[XY17] Yangyang Xu and Wotao Yin. A globally convergent algorithm for nonconvex optimization based on block coordinate update. *Journal of Scientific Computing*, 72(2):700–734, 2017. [Pages 12, 53, and 65.]

[YDC15] Alp Yurtsever, Quoc Tran Dinh, and Volkan Cevher. A universal primal-dual convex optimization framework. In *Advances in Neural Information Processing Systems*, pages 3150–3158, 2015. [Page 15.]

[YFLC18] Alp Yurtsever, Olivier Fercoq, Francesco Locatello, and Volkan Cevher. A conditional gradient framework for composite convex minimization with applications to semidefinite programming. *arXiv preprint arXiv:1804.08544*, 2018. [Pages 8, 15, 17, 46, and 49.]

[YST15] Liuqin Yang, Defeng Sun, and Kim-Chuan Toh. Sdpnal+: a majorized semismooth newton-cg augmented lagrangian method for semidefinite programming with nonnegative constraints. *Mathematical Programming Computation*, 7(3):331–366, 2015. [Page 17.]

[YTF+21] Alp Yurtsever, Joel A Tropp, Olivier Fercoq, Madeleine Udell, and Volkan Cevher. Scalable semidefinite programming. *SIAM Journal on Mathematics of Data Science*, 3(1):171–200, 2021. [Page 1.]

[YUTC17] Alp Yurtsever, Madeleine Udell, Joel Tropp, and Volkan Cevher. Sketchy decisions: Convex low-rank matrix optimization with optimal storage. In *Artificial intelligence and statistics*, pages 1188–1196. PMLR, 2017. [Page 1.]

[YY13] Junfeng Yang and Xiaoming Yuan. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of computation*, 82(281):301–329, 2013. [Page 52.]

[ZARX18] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, pages 9472–9483, 2018. [Pages 52 and 71.]

[Zei12] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. [Page 1.]

[ZGB+13] Bin Zhang, Chris Gaiteri, Liviu-Gabriel Bodea, Zhi Wang, Joshua McElwee, Alexei A Podtelezhnikov, Chunsheng Zhang, Tao Xie, Linh Tran, Radu Dobrin, et al. Integrated systems approach identifies genetic nodes and networks in late-onset alzheimer's disease. *Cell*, 153(3):707–720, 2013. [Page 71.]

[ZKRW98] Qing Zhao, Stefan E Karisch, Franz Rendl, and Henry Wolkowicz. Semidefinite programming relaxations for the quadratic assignment problem. *Journal of Combinatorial Optimization*, 2(1):71–109, 1998. [Pages 7 and 51.]

[ZYZ22] Jinshan Zeng, Wotao Yin, and Ding-Xuan Zhou. Moreau envelope augmented lagrangian method for nonconvex optimization with linear constraints. *Journal of Scientific Computing*, 91(2):61, 2022. [Page 2.]

MEHMET FATIH **SAHIN**

Lausanne, Switzerland +41 78 832 78 00
mehmetfatihsahin mehmetfatihsahin1@gmail.com

STRENGTHS

- ▷ Expert in optimization for Machine Learning
- ▷ Novel numerical algorithms and theoretical methods
- ▷ Design, implementation, optimization of codes
- ▷ Signal processing

EDUCATION

Ph.D. in Computer Science Feb. 2018 – Sept. 2022 (*expected*)
École Polytechnique Fédérale de Lausanne (EPFL) Lausanne, Switzerland

Thesis: *Augmented Lagrangian methods for provable and scalable machine learning*
Advisor: Prof. Volkan Cevher

Bachelor of Science in Electrical and Electronics Engineering Sept. 2012 – June 2017
Middle East Technical University | CGPA: 3.83/4.0 Ankara, Turkey

Bachelor of Science in Physics Sept. 2014 – June 2017
Middle East Technical University | CGPA: 3.87/4.0 Ankara, Turkey

CORE EXPERIENCE

Laboratory for Information and Inference Systems, EPFL Mar. 2018 – Present
Machine Learning Researcher and Ph.D. candidate

- Developed augmented Lagrangian-based algorithms for large scale machine learning (ML) problems
- Provided theoretical grounds for solving nonconvex constrained optimization problems
- Verified the theoretical findings through simulations for multiple ML problems (clustering, MaxCut) on Python
- Worked on a deep-learning based sampling mask design project for fast MRI acquisition
- Worked on efficient implementation of the regularized optimal transport problem
- Helped developing an algorithm for verification of polynomial neural networks based on nonlinear optimization
- Co-supervised a master thesis on deep reinforcement learning for optimal scheduling of hydropower plants
- Served as TA for different Master level courses: Reinforcement Learning, Deep Learning and Optimization

Laboratory for Information and Inference Systems, EPFL Sept. 2017– Mar. 2018
Machine Learning Research Intern

- Helped developing a no-regret optimistic mirror-descent algorithm for zero sum games
- Worked on developing projection-free adaptive optimization methods
- Worked on a semidefinite programming (SDP)-based blind deconvolution problem
- Extracted the graph adjacency matrix of Swiss rail network from image data using matched filters

ASELSAN Inc. Jan. 2017 – Sept. 2017
Signal Processing Engineer

- Design and optimization of beamforming algorithms for air and underwater acoustic sensor arrays
- Simulation of signal processing unit of gunshot detection system on MATLAB
- Carried out the field tests of the designed electronic products

PROGRAMMING SKILLS

Python (PyTorch, NumPy, SciPy, Matplotlib, Pandas), C, MATLAB, slurm, Git, Latex

LANGUAGES

English (**fluent**), French (**elementary**), Turkish (**native**)

Additional Experience

Middle East Technical University Sept. 2016 – Jun. 2017
Team leader at capstone design project
- Electronic and algorithmic design of a fully autonomous robot using Raspberry Pi (Video)

TUBITAK, Defense Industries Research and Development Institute June 2015 – July 2015
Electronics Engineering Intern
- Design of an SPI communication module for FPGA using VHDL

SANEL Auto Electronic July 2015 – Aug. 2015
Electronics Engineering Intern
- Embedded software design of **Microchip** microcontrollers used in automotive electronic systems

Honors and awards

Travel award, NeurIPS	2019
Volunteer award, ICML	2019
Best project award for machine learning course	2018
Honorable Mention award from graduation design project fair of METU	2017
Dean's High Honor Roll, METU	2013–2017
TUBITAK undergraduate scholarship	2013–2017
Ranked 901st at the university entrance exam among 2.5 million students	2013
School representative in National Physics Olympiad, TUBITAK	2011

Invited talks

SIAM Conference on Optimization July, 2021
Algorithms and complexity results for large-scale constrained optimization

EUROPT International workshop on continuous optimization July, 2021
Nonlinear composite and constrained optimization

Extracurricular activities

Turkish folk dancer at METU Turkish folklore society (3 years)
Organizing committee member of a nationwide startup competition at IEEE METU (2 years)
Volunteer guide for national science olymiapds at TUBITAK

Personal Information

28, married, Turkish citizen. Discharged from military service.

Publications

[1] **Sahin, M.F.**, A. Eftekhari, and V. Cevher. A linearized alternating direction method of multipliers framework for nonconvex problems with nonlinear constraints. *Work under review*.

[2] **Sahin, M.F.**, A. Eftekhari, A. Alacaoglu, F.L. Gómez, and V. Cevher. An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints. In *NeurIPS*, 2019.

[3] E.A. Kangarshahi, Y.P. Hsieh, **Sahin, M.F.**, and V. Cevher. Lets be honest: An optimal no-regret framework for zero-sum games. In *International Conference on Machine Learning*, pages 2488–2496. PMLR, 2018.

[4] B.C. Vu, A. Alacaoglu, **Sahin, M.F.**, A. Yurtsever, and V. Cevher. A first-order augmented lagrangian framework for nonconvex optimization with nonlinear constraints. *Workshop on Modern Trends in Nonconvex Optimization for Machine Learning, ICML.*, 2018.