An Inexact Augmented Lagrangian Framework for Non-Convex Optimization with Nonlinear Constraints

Mehmet Fatih Sahin [1] **Armin Eftekhari** [1] **Ahmet Alacaoglu** [1] **Fabian Latorre** [1] **Volkan Cevher** [1]

Abstract

We propose a practical inexact augmented Lagrangian method (iALM) for nonconvex problems with nonlinear constrains. We characterize the total computational complexity of our method subject to a verifiable geometric condition.

In particular, when a first-order solver is used for the inner iterates, we prove that iALM finds a first-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^3)$ calls to the first-order oracle. Likewise, when a second-order solver is used for the inner iterates, we prove that iALM finds a second-order stationary point with $\tilde{\mathcal{O}}(1/\epsilon^5)$ calls to the second-order oracle. These complexity results match the known theoretical results in the literature with a simple, implementable and versatile algorithm.

We provide numerical evidence on large-scale machine learning problems, including the Burer-Monteiro factorization of standard form Semidefinite Programming (SDP) relaxations, for which we verify our geometric condition in specific cases. For these problems and under suitable assumptions, our algorithm in fact achieves global optimality for the underlying convex SDP.

1. Introduction

We study the following nonconvex optimization problem

$$\begin{cases} \min\limits_{x\in\mathbb{R}^d} f(x)+g(x) \\ A(x)=b, \end{cases} \tag{1}$$

where $f:\mathbb{R}^d\to\mathbb{R}$ is possibly non-convex and $A:\mathbb{R}^d\to\mathbb{R}^m$ is a nonlinear operator and $b\in\mathbb{R}^m$. For clarity of notation, we take $b=0$ in the sequel, the extension to any b is trivial. We assume that $g:\mathbb{R}^d\to\mathbb{R}$ is proximal-friendly (possibly nonsmooth) convex function.

[1]LIONS, Ecole Polytechnique Fédérale de Lausanne, Switzerland. Correspondence to: Mehmet Fatih Sahin <mehmet.sahin@epfl.ch>.

Under Review.

A host of problems in computer science (Khot & Naor, 2011; Lovász, 2003), machine learning (Mossel et al., 2015; Song et al., 2007), and signal processing (Singer, 2011; Singer & Shkolnisky, 2011) naturally fall under the template of (1), including max-cut, clustering, generalized eigenvalue, as well as community detection.

To address these applications, this paper builds up on the vast literature on the classical inexact augmented Lagrangian framework and proposes a simple, intuitive as well as easy-to-implement algorithm with total complexity results for (1) under an interpretable geometric condition. Before we elaborate on the results, let us first motivate (1) with an important application to semidefinite programming (SDP):

Vignette: Burer-Monteiro splitting. A powerful convex relaxation for max-cut, clustering, and several other problems above is provided by the SDP

$$\begin{cases} \min\limits_{X\in\mathbb{S}^{d\times d}} \langle C,X\rangle \\ B(X)=b,\ X\succeq 0, \end{cases} \tag{2}$$

where $C\in\mathbb{R}^{d\times d}$ and X is a positive semidefinite and symmetric $d\times d$ matrix, and $B:\mathbb{S}^{d\times d}\to\mathbb{R}^m$ is a linear operator. If the unique-games conjecture is true, SDPs achieve the best approximation for the underlying discrete problem (Raghavendra, 2008).

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 therefore is to solve SDP's in small space and in a scalable fashion. A recent algorithm, i.e., homotopy conditional gradient method (HCGM) based on Linear Minimization Oracles (LMO), can address this template in small space via sketching (Yurtsever et al., 2018); however, such LMO-based methods are extremely slow in obtaining accurate solutions.

A key approach for solving (1), dating back to (Burer & Monteiro, 2003; 2005), is the so-called Burer-Monteiro (BR) splitting $X=UU^\top$, where $U\in\mathbb{R}^{d\times r}$ and r is selected according to the guidelines in (Pataki, 1998; Barvinok, 1995). It has been shown that these bounds on the rank, which are

shown to be optimal (Waldspurger & Waters, 2018), under some assumptions removing the spurious local minima of the nonconvex factorized problem (Boumal et al., 2016b).

This splitting results in the following non-convex problem

$$\begin{cases} \min\limits_{U\in\mathbb{R}^{d\times r}} \langle C, UU^\top\rangle \\ B(UU^\top) = b, \end{cases} \tag{3}$$

which can be written in the form of (1).

To solve (3), the inexact Augmented Lagrangian Method (iALM) is widely used (Burer & Monteiro, 2003; 2005; Kulis et al., 2007), due to its cheap per iteration cost and also its empirical success in practice. Every (outer) iteration of iALM calls a solver to inexactly solve an intermediate augmented Lagrangian subproblem to near stationarity, and the user has freedom in choosing this solver, which could use first-order (say, proximal gradient descent (Parikh et al., 2014)) or second-order information, such as BFGS (Nocedal & Wright, 2006).

We argue that unlike its convex counterpart (Nedelcu et al., 2014; Lan & Monteiro, 2016; Xu, 2017), the convergence rate and the complexity of iALM for (3) are not well-understood, see Section 5 for a review of the related literature. Indeed, addressing this important theoretical gap is one of the contributions of our work.

A brief summary of our contributions:

∘ Our framework is future-proof in the sense that we obtain the convergence rate of iALM for (1) with an arbitrary solver for finding first- and second-order stationary points.

∘ We investigate using different solvers for augmented Lagrangian subproblems and provide overall iteration complexity bounds for finding first- and second-order stationary points of (1). Our complexity bounds match the best theoretical complexity results in optimization, see Section 5.

∘ We propose a novel geometric condition that simplifies the algorithmic analysis for iALM. We verify the condition for key problems described in Section 6.

Roadmap. Section 2 collects the main tools and our notation. We present the iALM in Section 3 and obtain its convergence rate to first- and second-order stationary points in Section 4, alongside their iteration complexities. We provide a comprehensive review of the literature and highlight our key differences in Section 5. Section 6 presents the numerical evidence and comparisons with the state-of-the-art.

2. Preliminaries

Notation. We use the notation $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ for the standard inner product and the norm on $\mathbb{R}^d$. For matrices, $\|\cdot\|$ and $\|\cdot\|_F$ denote the spectral and the Frobenius norms, respectively. For a 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.

We use the indicator function $\delta_{\mathcal{X}}:\mathbb{R}^d\to\mathbb{R}$ of a set $\mathcal{X}\subset\mathbb{R}^d$, which takes x to

$$\delta_{\mathcal{X}}(x) = \begin{cases} 0 & x\in\mathcal{X} \\ \infty & x\notin\mathcal{X}. \end{cases} \tag{4}$$

The distance function from a point x to $\mathcal{X}$ is denoted by $\mathrm{dist}(x,\mathcal{X}) = \min_{z\in\mathcal{X}}\|x-z\|$. For integers $k_0\le k_1$, we denote $[k_0:k_1]=\{k_0,\dots,k_1\}$.

For an operator $A:\mathbb{R}^d\to\mathbb{R}^m$ with components $\{A_i\}_{i=1}^m$, we let $DA(x)\in\mathbb{R}^{m\times d}$ denote the Jacobian of A, where the ith row of $DA(x)$ is the gradient vector $\nabla A_i(x)\in\mathbb{R}^d$.

Smoothness. We require $f:\mathbb{R}^d\to\mathbb{R}$ and $A:\mathbb{R}^d\to\mathbb{R}^m$ in (1) to be smooth; i.e., there exists $\lambda_f,\lambda_A\ge 0$ such that

$$\begin{aligned} \|\nabla f(x)-\nabla f(x')\| &\le \lambda_f\|x-x'\|, \\ \|DA(x)-DA(x')\| &\le \lambda_A\|x-x'\|, \end{aligned} \tag{5}$$

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

The augmented Lagrangian method (ALM). ALM is a classical algorithm, first appeared in (Hestenes, 1969; Powell, 1969) and extensively studied in (Bertsekas, 2014). For solving (1), ALM suggests solving the problem

$$\min_x \max_y \ \mathcal{L}_\beta(x,y) + g(x), \tag{6}$$

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

$$\mathcal{L}_\beta(x,y) := f(x) + \langle A(x), y\rangle + \frac{\beta}{2}\|A(x)\|^2. \tag{7}$$

The minimax formulation in (6) naturally suggests the following algorithm for solving (1). For dual step sizes $\{\sigma_k\}_k$, consider the iterates

$$\begin{aligned} x_{k+1} &\in \operatorname*{argmin}_x \ \mathcal{L}_\beta(x,y_k) + g(x), \\ y_{k+1} &= y_k + \sigma_k A(x_{k+1}). \end{aligned} \tag{8}$$

However, updating x_{k+1} above requires solving the nonconvex problem (8) to optimality, which is typically intractable. Instead, it is often easier to find an approximate first- or second-order stationary point of (8).

Hence, we argue that by gradually improving the stationarity precision and increasing the penalty weight β above, we can reach a stationary point of the main problem in (1), as detailed in Section 3.

Optimality conditions. First-order necessary optimality conditions for (1) are well-understood. Indeed, $x \in \mathbb{R}^d$ is a first-order stationary point of (1) if there exists $y \in \mathbb{R}^m$ such that

$$\begin{cases} -\nabla f(x) - DA(x)^\top y \in \partial g(x) \\ A(x) = 0, \end{cases} \tag{9}$$

where $DA(x)$ is the Jacobian of A at x. Recalling (7), we observe that (9) is equivalent to

$$\begin{cases} -\nabla_x \mathcal{L}_\beta(x, y) \in \partial g(x) \\ A(x) = 0, \end{cases} \tag{10}$$

which is in turn the necessary optimality condition for (6). Inspired by this, we say that x is an (ϵ_f, β) first-order stationary point if

$$\begin{cases} \text{dist}(-\nabla_x \mathcal{L}_\beta(x, y), \partial g(x)) \le \epsilon_f \\ \|A(x)\| \le \epsilon_f, \end{cases} \tag{11}$$

for $\epsilon_f \ge 0$. In light of (11), a suitable metric for evaluating the stationarity of a pair $(x, y) \in \mathbb{R}^d \times \mathbb{R}^m$ is

$$\text{dist}\left(-\nabla_x \mathcal{L}_\beta(x, y), \partial g(x)\right) + \|A(x)\|, \tag{12}$$

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

$$\text{dist}\left(u, \partial g(x)\right) = \|P_{T_{\mathcal{X}}(x)}(u)\|. \tag{13}$$

When $g = 0$, a first-order stationary point $x \in \mathbb{R}^d$ of (1) is also second-order stationary if

$$\lambda_{\min}(\nabla_{xx} \mathcal{L}_\beta(x, y)) > 0, \tag{14}$$

where $\nabla_{xx} \mathcal{L}_\beta$ is the Hessian 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 (11), it holds that

$$\lambda_{\min}(\nabla_{xx} \mathcal{L}_\beta(x, y)) \ge -\epsilon_s, \tag{15}$$

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

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

Lemma 2.1 (smoothness). *For fixed $y \in \mathbb{R}^m$ and $\rho, \rho' \ge 0$, it holds that*

$$\|\nabla_x \mathcal{L}_\beta(x, y) - \nabla_x \mathcal{L}_\beta(x', y)\| \le \lambda_\beta \|x - x'\|, \tag{16}$$

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

$$\begin{aligned} \lambda_\beta &\le \lambda_f + \sqrt{m}\lambda_A \|y\| + (\sqrt{m}\lambda_A \rho + d\lambda_A'^2)\beta \\ &=: \lambda_f + \sqrt{m}\lambda_A \|y\| + \lambda''(A, \rho, \rho')\beta. \end{aligned} \tag{17}$$

Above, λ_f, λ_A were defined in (5) and

$$\lambda_A' := \max_{\|x\| \le \rho'} \|DA(x)\|. \tag{18}$$

3. Our optimization framework

To solve the formulation presented in (6), we propose the inexact ALM (iALM), detailed in Algorithm 1.

At the iteration k, Algorithm 1 calls in Step 2 a solver that finds an approximate stationary point of the augmented Lagrangian $\mathcal{L}_{\beta_k}(\cdot, y_k)$ 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 (1). As we will see in the convergence analysis, the particular choice of the dual step size σ_k in Algorithm 1 ensures that the dual variable y_k remains bounded; see (Bertsekas, 1976) for a precedent in the ALM literature where a similar choice for σ_k is considered.

Step 3 of Algorithm 1 removes pathological cases with divergent iterates. As an example, suppose that $g = \delta_{\mathcal{X}}$ in (1) is the indicator function for a bounded convex set $\mathcal{X} \subset \mathbb{R}^d$ and take $\rho' > \max_{x \in \mathcal{X}} \|x\|$. Then, for sufficiently large k, it is not difficult to verify that all the iterates of Algorithm 1 automatically satisfy $\|x_k\| \le \rho'$ without the need to execute Step 3.

4. Convergence Rate

In this section, we detail the convergence rate of Algorithm 1 for finding first-order and second-order stationary points, along with the iteration complexity results. All the proofs are deferred to Appendix A for the clarity.

Theorem 4.1 below characterizes the convergence rate of Algorithm 1 for finding stationary points in terms of the number of outer iterations.

Theorem 4.1. (convergence rate) *Suppose that f and A are smooth in the sense specified in (5). For $\rho' > 0$, let*

$$\lambda_f' = \max_{\|x\| \le \rho'} \|\nabla f(x)\|, \quad \lambda_A' = \max_{\|x\| \le \rho'} \|DA(x)\|, \tag{19}$$

be the (restricted) Lipschitz constants of f and A, respectively. For sufficiently large integers $k_0 \le k_1$, 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. For $\nu > 0$,

Algorithm 1 Inexact AL for solving (1)

Input: $\rho, \rho', \rho'' > 0$. A non-decreasing, positive, unbounded sequence $\{\beta_k\}_{k\geq 1}$, stopping thresholds τ_f and τ_s.

Initialization: $x_1 \in \mathbb{R}^d$ such that $\|A(x_1)\| \leq \rho$ and $\|x_1\| \leq \rho'$, $y_0 \in \mathbb{R}^m$, σ_1.

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

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

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

for first-order stationarity and, in addition,

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

for second-order-stationarity.

3. **(Control)** If necessary, project x_{k+1} to ensure that $\|x_{k+1}\| \leq \rho'$.
4. **(Update dual step size)**

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

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

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

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

end for

assume that

$$\nu\|A(x_k)\| \leq \operatorname{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right), \quad (20)$$

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) with*

$$\epsilon_{k,f} = \frac{1}{\beta_{k-1}}\left(\frac{2(\lambda'_f + \lambda'_A y_{\max})(1+\lambda'_A\sigma_k)}{\nu} + 1\right) =: \frac{Q(f, g, A, \sigma_1)}{\beta_{k-1}}, \quad (21)$$

for every $k \in K$, where the expression for $y_{\max} = y_{\max}(x_1, y_0, \sigma_1)$ is given in (43).

- *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) with $\epsilon_{k,s}$ specified above and with*

$$\epsilon_{k,s} = \epsilon_{k-1} + \sigma_k\sqrt{m}\lambda_A \frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu\beta_{k-1}} = \frac{\nu + \sigma_k\sqrt{m}\lambda_A 2\lambda'_f + 2\lambda'_A y_{\max}}{\nu\beta_{k-1}} =: \frac{Q'}{\beta_{k-1}}. \quad (22)$$

Loosely speaking, Theorem 4.1 states that Algorithm 1 converges to a (first- or second-) order stationary point of (1) at the rate of $1/\beta_k$.

A few remarks are in order.

Regularity. The key condition in Theorem 4.1 is (20) which, broadly speaking, controls the problem geometry.

As the penalty weight β_k grows, the primal solver in Step 2 places an increasing emphasis on reducing the feasibility gap and (20) formalizes this intuition. In contrast to most conditions in the nonconvex optimization literature, such as (Flores-Bazán et al., 2012), our condition in (20) appears to be easier to verify, as we see in Section 6.

We now argue that such a condition is necessary for controlling the feasibility gap of (1) and ensuring the success of Algorithm 1. Consider the case where $f = 0$ and $g = \delta_{\mathcal{X}}$, where $\mathcal{X}$ is a convex set, A is a linear operator. In this case, solving (1) finds a point in $\mathcal{X} \cap \operatorname{null}(A)$, where the subspace $\operatorname{null}(A) = \{x \in \mathbb{R}^d : A(x) = 0\} \subset \mathbb{R}^d$ is the null space of A. Here, the Slater's condition requires that

$$\operatorname{relint}(\mathcal{X}) \cap \operatorname{null}(A) \neq \emptyset. \quad (23)$$

In general, the Slater's condition plays a key role in convex optimization as a sufficient condition for strong duality and, as a result, guarantees the success of a variety of primal-dual algorithms for linearly-constrained convex programs (Bauschke et al., 2011).

Intuitively, the Slater's condition here removes any pathological cases by ensuring that the subspace $\operatorname{null}(A)$ is not tangent to $\mathcal{X}$, see Figure 1. In such pathological cases, solving (1), namely, finding a point in $\mathcal{X} \cap \operatorname{null}(A)$, can be particularly difficult. For instance, the alternating projection algorithm (which iteratively projects onto $\mathcal{X}$ and $\operatorname{null}(A)$) has arbitrarily slow convergence, see Figure 1.

Computational complexity. Theorem 4.1 allows us to specify the number of iterations that Algorithm 1 requires to reach a near-stationary point of Program (1) with a prescribed precision and, in particular, specifies the number of calls made to the solver in Step 2. In this sense, Theorem 4.1

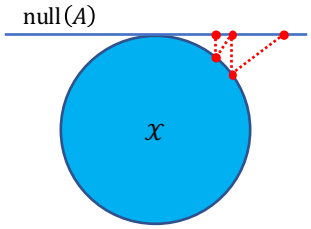

Figure 1. Solving (1) can be particularly difficult, even when (1) is a convex program. We present a pathological geometry where the Slater's condition does not apply. See the first remark after Theorem 4.1 for more details.

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 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 (Ghadimi & Lan, 2016). In the second scenario, we will use the second-order trust region method developed in (Cartis et al., 2012).

4.1. First-Order Optimality

Let us first consider the first-order optimality case where the solver in Step 2 is APGM (Ghadimi & Lan, 2016). APGM makes use of $\nabla_x \mathcal{L}_\beta(x,y)$, prox_g and classical Nesterov acceleration step for the iterates (Nesterov, 1983) to obtain first order stationarity guarantees for solving (8). Suppose that $g = \delta_{\mathcal{X}}$ is the indicator function on a bounded convex set $\mathcal{X} \subset \mathbb{R}^d$ and let

$$x_{\max} = \max_{x \in \mathcal{X}} \|x\|, \tag{24}$$

be the radius of a ball centered at the origin that includes $\mathcal{X}$. Then, adapting the results in (Ghadimi & Lan, 2016) to our setup, APGM reaches x_k in Step 2 of Algorithm 1 after

$$\mathcal{O}\left(\frac{\lambda_{\beta_k}^2 x_{\max}^2}{\epsilon_{k+1}}\right) \tag{25}$$

(inner) iterations, where λ_{β_k} denotes the Lipschitz constant of $\nabla_x \mathcal{L}_{\beta_k}(x,y)$, bounded in (17). We note that for simplicity, we use a looser bound in (25) than (Ghadimi & Lan, 2016). Using (25), we derive the following corollary, describing the total computational complexity of Algorithm 1 in terms of the calls to the first-order oracle in APGM.

Corollary 4.2. *For $b > 1$, let $\beta_k = b^k$ for every k. If we use APGM from (Ghadimi & Lan, 2016) for Step 2 of Algorithm 1, the algorithm finds an (ϵ_f, β_k) first-order stationary point, see (11), after T calls to the first-order oracle, where*

$$T = \mathcal{O}\left(\frac{Q^3 x_{\max}^2}{\epsilon^3} \log_b\left(\frac{Q}{\epsilon}\right)\right) = \tilde{\mathcal{O}}\left(\frac{Q^3 x_{\max}^2}{\epsilon^3}\right). \tag{26}$$

For Algorithm 1 to reach a near-stationary point with an accuracy of ϵ_f in the sense of (11) 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 (21) in Theorem 4.1. In general, however, the constants in (21) are unknown and this approach is intractable. 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 4.2.

4.2. 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 (Cartis et al., 2012). Trust region method minimizes 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 (Nouiehed et al., 2018), 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 (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 (Cartis et al., 2012) to our setup, we find that the number of (inner) iterations required in Step 2 of Algorithm 1 to produce x_{k+1} is

$$\mathcal{O}\left(\frac{\lambda_{\beta_k,H}^2 (\mathcal{L}_{\beta_k}(x_1,y) - \min_x \mathcal{L}_{\beta_k}(x,y))}{\epsilon_k^3}\right), \tag{27}$$

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 2.1 and x_1 is the initial iterate of the given outer loop. In (Cartis et al., 2012), the term $\mathcal{L}_\beta(x_1,y) - \min_x \mathcal{L}_\beta(x,y)$ is bounded by a constant independent of ϵ. We assume a uniform bound for this quantity $\forall \beta_k$, instead of for one value of β_k as in (Cartis et al., 2012). Using (27) and Theorem 4.1, we arrive at the following:

Corollary 4.3. *For $b > 1$, let $\beta_k = b^k$ for every k. We assume that*

$$\mathcal{L}_\beta(x_1,y) - \min_x \mathcal{L}_\beta(x,y) \leq L_u, \quad \forall \beta. \tag{28}$$

If we use the trust region method from (Cartis et al., 2012) for Step 2 of Algorithm 1, the algorithm finds an ϵ-second-order stationary point of (1) in T calls to the second-order oracle where

$$T \leq \mathcal{O}\left(\frac{L_u Q'^5}{\epsilon^5}\log_b\left(\frac{Q'}{\epsilon}\right)\right) = \widetilde{\mathcal{O}}\left(\frac{L_u Q'^5}{\epsilon^5}\right). \quad (29)$$

Before closing this section, we note that the remark after Corollary 4.2 applies here as well.

5. Related works

ALM has a long history in the optimization literature, dating back to (Hestenes, 1969; Powell, 1969). In the special case of (1) with a convex function f and a linear operator A, standard, inexact and linearized versions of ALM have been extensively studied (Lan & Monteiro, 2016; Nedelcu et al., 2014; Tran-Dinh et al., 2018; Xu, 2017).

Classical works on ALM focused on the general template of (1) with nonconvex f and nonlinear A, with arguably stronger assumptions and required exact solutions to the subproblems of the form (8), which appear in Step 2 of Algorithm 1, see for instance (Bertsekas, 2014).

A similar analysis was conducted in (Fernández & Solodov, 2012) for the general template of (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, unlike our results, the authors did not analyze how to solve the subproblems inexactly and they did not provide total complexity results and verifiable conditions.

Problem (1) with similar assumptions to us is also studied in (Birgin et al., 2016) and (Cartis et al., 2018) for first-order and second-order stationarity, respectively, with explicit iteration complexity analysis. As we have mentioned in Section 4, our iteration complexity results 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. In contrast to (Birgin et al., 2016; Cartis et al., 2018), Algorithm 1 does not set accuracies a priori.

(Cartis et al., 2011) also considers the same template (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 match our complexity results.

(Bolte et al., 2018) studies the general template (1) with specific assumptions involving local error bound conditions for the (1). These conditions are studied in detail in (Bolte et al., 2017), but their validity for general SDPs (2) has never been established. This work also lacks the total iteration complexity analysis presented here.

Another work (Clason et al., 2018) focused on solving (1) by adapting the primal-dual method of Chambolle and Pock (Chambolle & Pock, 2011). 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.

(Burer & Monteiro, 2003; 2005) is the first work that proposes the splitting $X = UU^\top$ for solving SDPs of the form (2). Following these works, the literature on Burer-Monteiro (BM) splitting for the large part focused on using ALM for solving the reformulated problem (3).

However, this approach 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.

(Bhojanapalli et al., 2016; Park et al., 2016) 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) where the difficulty arises due to the linear constraints.

(Bhojanapalli et al., 2018) focused on the quadratic penalty formulation of (1), namely,

$$\min_{X\succeq 0}\langle C, X\rangle + \frac{\mu}{2}\|B(x) - b\|^2, \quad (30)$$

which after BM splitting becomes

$$\min_{U\in\mathbb{R}^{d\times r}}\langle C, UU^\top\rangle + \frac{\mu}{2}\|B(UU^\top) - b\|^2, \quad (31)$$

for which they study the optimality of the second-order stationary points. These results are for establishing a connection between the stationary points of (31) and global optima of (30). In contrast, we focus on the relation of the stationary points of (6) to the constrained problem (1).

Another popular method for solving SDPs are due to (Boumal et al., 2014; 2016a;b), focusing on the case where the constraints in (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 (Boumal et al., 2016a), per iteration cost of their method for max-cut problem is an astronomical $\mathcal{O}(d^6)$.

Lastly, there also exists a line of work for solving SDPs in their original convex formulation, in a storage efficient way (Nesterov, 2009; Yurtsever et al., 2015; 2018). 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 nonconvex approaches (Jaggi, 2013).

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 (Dai, 2002; Mascarenhas, 2004). For this reason, we have used the trust region method as the second-order solver in our analysis in Section 4, which is well-studied for non-convex problems (Cartis et al., 2012).

Empirically, however, BFGS and lBGFS are extremely successful and we have also opted for those solvers in this section since the subroutine does not affect Theorem 4.1 as long as the subsolver can perform in practice.

6.1. k-Means 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 (Peng & Wei, 2007), the authors provide a SDP relaxation of the clustering problem, specified as

$$\begin{cases} \min\limits_{Y\in\mathbb{R}^{n\times n}} \operatorname{tr}(DY) \\ Y\mathbf{1} = \mathbf{1},\ \operatorname{tr}(Y) = k,\ Y \succeq 0,\ Y \geq 0, \end{cases} \tag{32}$$

where k 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 Burer-Monteiro splitting, sacrificing convexity to reduce the computational complexity. More specifically, we solve the program

$$\begin{cases} \min\limits_{V\in\mathbb{R}^{n\times r}} \operatorname{tr}(DVV^\top) \\ VV^\top\mathbf{1} = \mathbf{1},\ \ \|V\|_F^2 \leq k,\ \ V \geq 0, \end{cases} \tag{33}$$

where $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. Note that $Y \geq 0$ in (32) is replaced above by the much stronger but easier to enforce constraint $V \geq 0$ constraint above, see (Kulis et al., 2007) for the reasoning behind this relaxation. Now, we can cast (33) as an instance of (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

$$f(x) = \sum_{i,j=1}^n D_{i,j} \langle x_i, x_j \rangle, \qquad g = \delta_C,$$

$$A(x) = [x_1^\top \sum_{j=1}^n x_j - 1, \cdots, x_n^\top \sum_{j=1}^n x_j - 1]^\top, \tag{34}$$

where C is the intersection of the positive orthant in $\mathbb{R}^d$ with the Euclidean ball of radius $\sqrt{k}$. In Appendix E, we somewhat informally verify that Theorem 4.1 applies to (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 (8) with convergence guarantees to first-order stationarity, as discussed in Section 4. lBFGS is a limited-memory version of BFGS algorithm in (Fletcher, 2013) 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(Yurtsever et al., 2018) which directly solves (32).
- SDPNAL+: A second-order augmented Lagrangian method for solving SDP's with nonnegativity constraints (Yang et al., 2015).

As for the dataset, our experimental setup is similar to that described by (Mixon et al., 2016). We use the publicly-available fashion-MNIST data in (Xiao et al., 2017), 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

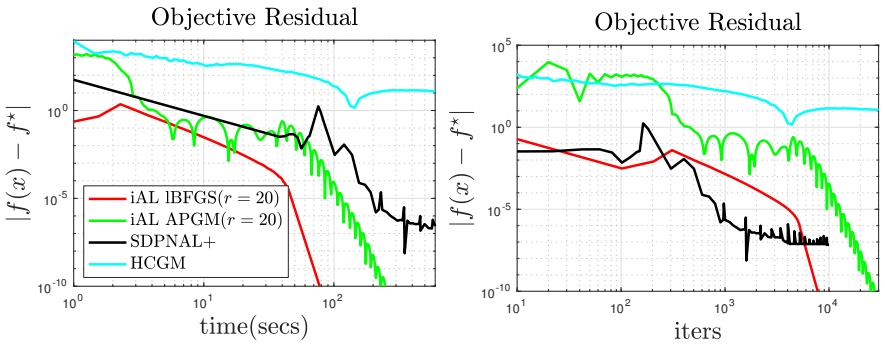

Figure 2. Convergence of different algorithms for k-Means clustering with fashion MNIST dataset. Here, we set the rank to be equal to 20 for the non-convex approaches. The solution rank for the template (32) is known and it is equal to number of clusters k (Theorem 1. (Kulis et al., 2007)). As discussed in (Tepper et al., 2018), setting rank $r > k$ leads more accurate reconstruction in expense of speed. Therefore, we set the rank to 20.

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 results are depicted in Figure 2. We implemented 3 algorithms on MATLAB and used the software package for SDPNAL+ which contains mex files. Performance of the nonconvex approach would be much better if we also used mex files.

6.2. Basis Pursuit

Basis Pursuit (BP) finds sparsest solutions of an underdetermined system of linear equations, namely,

$$\begin{cases} \min_z \|z\|_1 \\ Bz = b, \end{cases} \tag{35}$$

where $B \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$. BP has found many applications in machine learning, statistics and signal processing (Chen et al., 2001; Candès & Wakin, 2008; Arora et al., 2018). A huge number of primal-dual convex optimization algorithms are proposed to solve BP, including, but not limited to (Tran-Dinh et al., 2018; Chambolle & Pock, 2011). There also exists many line of works (Beck & Teboulle, 2009) to handle sparse regression problem via regularization with ℓ_1 norm.

Here, we take a different approach and cast (35) as an instance of (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, (35) is equivalent to (1) with

$$\begin{aligned} f(x) =& \|x\|_2^2, \quad g(x) = 0 \\ A(x) =& \overline{B}x^{\circ 2} - b. \end{aligned} \tag{36}$$

In Appendix D, we verify with minimal detail that Theorem 4.1 indeed applies to (1) with the above f, A.

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 $\sigma^2 = 10^{-6}$.

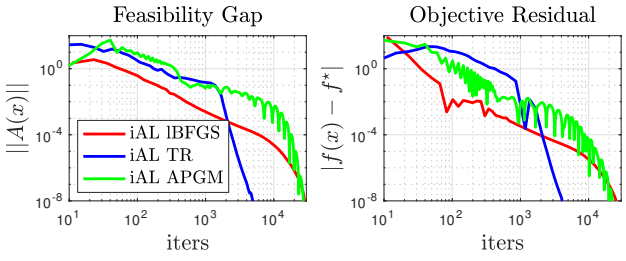

Figure 3. Convergence with different subsolvers for the aforementioned non-convex relaxation.

Figure 3 compiles our results for the proposed relaxation. It is, indeed, interesting to see that these type of non-convex relaxations gives the solution of convex one and first order methods succeed.

6.3. Adversarial Denoising with GANs

In the appendix, we provide a contemporary application example that our template applies.

References

Arora, S., Khodak, M., Saunshi, N., and Vodrahalli, K. A compressed sensing view of unsupervised text embeddings, bag-of-n-grams, and lstms. 2018.

Barvinok, A. I. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13(2):189–202, 1995.

Bauschke, H. H., Combettes, P. L., et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Bertsekas, D. P. On penalty and multiplier methods for constrained minimization. *SIAM Journal on Control and Optimization*, 14(2):216–235, 1976.

Bertsekas, D. P. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

Bhojanapalli, S., Kyrillidis, A., and Sanghavi, S. Dropping convexity for faster semi-definite optimization. In *Conference on Learning Theory*, pp. 530–582, 2016.

Bhojanapalli, S., Boumal, N., Jain, P., and Netrapalli, P. Smoothed analysis for low-rank solutions to semidefinite programs in quadratic penalty form. *arXiv preprint arXiv:1803.00186*, 2018.

Birgin, E. G., Gardenghi, J., Martinez, J. M., Santos, S., and Toint, P. L. Evaluation complexity for nonlinear constrained optimization using unscaled kkt conditions and high-order models. *SIAM Journal on Optimization*, 26(2):951–967, 2016.

Bolte, J., Nguyen, T. P., Peypouquet, J., and Suter, B. W. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming*, 165(2):471–507, 2017.

Bolte, J., Sabach, S., and Teboulle, M. Nonconvex lagrangian-based optimization: monitoring schemes and global convergence. *Mathematics of Operations Research*, 2018.

Boumal, N., Mishra, B., Absil, P.-A., and Sepulchre, R. Manopt, a matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research*, 15(1):1455–1459, 2014.

Boumal, N., Absil, P.-A., and Cartis, C. Global rates of convergence for nonconvex optimization on manifolds. *arXiv preprint arXiv:1605.08101*, 2016a.

Boumal, N., Voroninski, V., and Bandeira, A. The non-convex burer-monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems*, pp. 2757–2765, 2016b.

Burer, S. and Monteiro, R. D. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

Burer, S. and Monteiro, R. D. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.

Candès, E. J. and Wakin, M. B. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2): 21–30, 2008.

Cartis, C., Gould, N. I., and Toint, P. L. On the evaluation complexity of composite function minimization with applications to nonconvex nonlinear programming. *SIAM Journal on Optimization*, 21(4):1721–1739, 2011.

Cartis, C., Gould, N. I., and Toint, P. L. Complexity bounds for second-order optimality in unconstrained optimization. *Journal of Complexity*, 28(1):93–108, 2012.

Cartis, C., Gould, N. I., and Toint, P. L. Optimality of orders one to three and beyond: characterization and evaluation complexity in constrained nonconvex optimization. *Journal of Complexity*, 2018.

Chambolle, A. and Pock, T. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011.

Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic decomposition by basis pursuit. *SIAM review*, 43(1): 129–159, 2001.

Clason, C., Mazurenko, S., and Valkonen, T. Acceleration and global convergence of a first-order primal–dual method for nonconvex problems. *arXiv preprint arXiv:1802.03347*, 2018.

Dai, Y.-H. Convergence properties of the bfgs algoritm. *SIAM Journal on Optimization*, 13(3):693–701, 2002.

Fernández, D. and Solodov, M. V. 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.

Fletcher, R. *Practical methods of optimization*. John Wiley & Sons, 2013.

Flores-Bazán, F., Flores-Bazán, F., and Vera, C. A complete characterization of strong duality in nonconvex optimization with a single constraint. *Journal of Global Optimization*, 53(2):185–201, 2012.

Ghadimi, S. and Lan, G. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Networks. *ArXiv e-prints*, June 2014. URL https://arxiv.org/abs/1406.2661.

Hestenes, M. R. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320, 1969.

Ilyas, A., Jalal, A., Asteri, E., Daskalakis, C., and Dimakis, A. G. The Robust Manifold Defense: Adversarial Training using Generative Models. *arXiv e-prints*, art. arXiv:1712.09196, December 2017.

Jaggi, M. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pp. 427–435, 2013.

Khot, S. and Naor, A. Grothendieck-type inequalities in combinatorial optimization. *arXiv preprint arXiv:1108.2464*, 2011.

Kulis, B., Surendran, A. C., and Platt, J. C. Fast low-rank semidefinite programming for embedding and clustering. In *Artificial Intelligence and Statistics*, pp. 235–242, 2007.

Lan, G. and Monteiro, R. D. Iteration-complexity of first-order augmented lagrangian methods for convex programming. *Mathematical Programming*, 155(1-2):511–547, 2016.

Lovász, L. Semidefinite programs and combinatorial optimization. In *Recent advances in algorithms and combinatorics*, pp. 137–194. Springer, 2003.

Mascarenhas, W. F. The bfgs method with exact line searches fails for non-convex objective functions. *Mathematical Programming*, 99(1):49–61, 2004.

Mixon, D. G., Villar, S., and Ward, R. Clustering subgaussian mixtures by semidefinite programming. *arXiv preprint arXiv:1602.06612*, 2016.

Mossel, E., Neeman, J., and Sly, A. Consistency thresholds for the planted bisection model. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pp. 69–75. ACM, 2015.

Nedelcu, V., Necoara, I., and Tran-Dinh, Q. Computational complexity of inexact gradient augmented lagrangian methods: application to constrained mpc. *SIAM Journal on Control and Optimization*, 52(5):3109–3134, 2014.

Nesterov, Y. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.

Nesterov, Y. E. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. Akad. Nauk SSSR*, volume 269, pp. 543–547, 1983.

Nocedal, J. and Wright, S. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006. ISBN 9780387400655. URL https://books.google.ch/books?id=VbHYoSyelFcC.

Nouiehed, M., Lee, J. D., and Razaviyayn, M. Convergence to second-order stationarity for constrained non-convex optimization. *arXiv preprint arXiv:1810.02024*, 2018.

Obozinski, G., Jacob, L., and Vert, J.-P. Group lasso with overlaps: the latent group lasso approach. *arXiv preprint arXiv:1110.0413*, 2011.

Parikh, N., Boyd, S., et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

Park, D., Kyrillidis, A., Bhojanapalli, S., Caramanis, C., and Sanghavi, S. Provable burer-monteiro factorization for a class of norm-constrained matrix problems. *arXiv preprint arXiv:1606.01316*, 2016.

Pataki, G. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998.

Peng, J. and Wei, Y. Approximating K–means–type clustering via semidefinite programming. *SIAM J. Optim.*, 18 (1):186–205, 2007.

Powell, M. J. A method for nonlinear constraints in minimization problems. *Optimization*, pp. 283–298, 1969.

Raghavendra, P. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 245–254. ACM, 2008.

Singer, A. Angular synchronization by eigenvectors and semidefinite programming. *Applied and computational harmonic analysis*, 30(1):20, 2011.

Singer, A. and Shkolnisky, Y. 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.

Song, L., Smola, A., Gretton, A., and Borgwardt, K. M. A dependence maximization view of clustering. In *Proceedings of the 24th international conference on Machine learning*, pp. 815–822. ACM, 2007.

Tepper, M., Sengupta, A. M., and Chklovskii, D. Clustering is semidefinitely not that hard: Nonnegative sdp for manifold disentangling. *Journal of Machine Learning Research*, 19(82), 2018.

Tran-Dinh, Q., Alacaoglu, A., Fercoq, O., and Cevher, V. An adaptive primal-dual framework for nonsmooth convex minimization. *arXiv preprint arXiv:1808.04648*, 2018.

Waldspurger, I. and Waters, A. Rank optimality for the burer-monteiro factorization. *arXiv preprint arXiv:1812.03046*, 2018.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Xu, Y. Iteration complexity of inexact augmented lagrangian methods for constrained convex programming. *arXiv preprint arXiv:1711.05812v2*, 2017.

Yang, L., Sun, D., and Toh, K.-C. Sdpnal+: a majorized semismooth newton-cg augmented lagrangian method for semidefinite programming with nonnegative constraints. *Mathematical Programming Computation*, 7(3):331–366, 2015.

Yurtsever, A., Dinh, Q. T., and Cevher, V. A universal primal-dual convex optimization framework. In *Advances in Neural Information Processing Systems*, pp. 3150–3158, 2015.

Yurtsever, A., Fercoq, O., Locatello, F., and Cevher, V. A conditional gradient framework for composite convex minimization with applications to semidefinite programming. *arXiv preprint arXiv:1804.08544*, 2018.

Appendix

A. Proof of Theorem 4.1

For every $k \geq 2$, recall from (7) and Step 2 of Algorithm 1 that x_k satisfies

$$\begin{aligned}&\operatorname{dist}(-\nabla f(x_k) - DA(x_k)^\top y_{k-1} \\ &\qquad - \beta_{k-1} DA(x_k)^\top A(x_k), \partial g(x_k)) \\ &= \operatorname{dist}(-\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_{k-1}), \partial g(x_k)) \leq \epsilon_k. \end{aligned} \tag{37}$$

With an application of the triangle inequality, it follows that

$$\begin{aligned}&\operatorname{dist}(-\beta_{k-1} DA(x_k)^\top A(x_k), \partial g(x_k)) \\ &\qquad \leq \|\nabla f(x_k)\| + \|DA(x_k)^\top y_{k-1}\| + \epsilon_k, \end{aligned} \tag{38}$$

which in turn implies that

$$\begin{aligned}&\operatorname{dist}(-DA(x_k)^\top A(x_k), \partial g(x_k)/\beta_{k-1}) \\ &\leq \frac{\|\nabla f(x_k)\|}{\beta_{k-1}} + \frac{\|DA(x_k)^\top y_{k-1}\|}{\beta_{k-1}} + \frac{\epsilon_k}{\beta_{k-1}} \\ &\leq \frac{\lambda_f' + \lambda_A' \|y_{k-1}\| + \epsilon_k}{\beta_{k-1}}, \end{aligned} \tag{39}$$

where λ_f', λ_A' were defined in (19). We next translate (39) into a bound on the feasibility gap $\|A(x_k)\|$. Using the regularity condition (20), the left-hand side of (39) can be bounded below as

$$\begin{aligned}&\operatorname{dist}(-DA(x_k)^\top A(x_k), \partial g(x_k)/\beta_{k-1}) \\ &\geq \nu \|A(x_k)\|, \qquad \text{(see (20))} \end{aligned} \tag{40}$$

provided that ρ, ρ' satisfy

$$\max_{k \in K} \|A(x_k)\| \leq \rho, \qquad \max_{k \in K} \|x_k\| \leq \rho'. \tag{41}$$

By substituting (40) back into (39), we find that

$$\|A(x_k)\| \leq \frac{\lambda_f' + \lambda_A' \|y_{k-1}\| + \epsilon_k}{\nu \beta_{k-1}}. \tag{42}$$

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

$$\begin{aligned}\|y_k\| &= \|y_0 + \sum_{i=1}^{k} \sigma_i A(x_i)\| \quad \text{(Step 5 of Algorithm 1)} \\ &\leq \|y_0\| + \sum_{i=1}^{k} \sigma_i \|A(x_i)\| \qquad \text{(triangle inequality)} \\ &\leq \|y_0\| + \sum_{i=1}^{k} \frac{\|A(x_1)\| \log^2 2}{k \log^2(k+1)} \quad \text{(Step 4)} \\ &\leq \|y_0\| + c\|A(x_1)\| \log^2 2 =: y_{\max}, \end{aligned} \tag{43}$$

where

$$c \geq \sum_{i=1}^{\infty} \frac{1}{k \log^2(k+1)}. \tag{44}$$

Substituting (43) back into (42), we reach

$$\begin{aligned}\|A(x_k)\| &\leq \frac{\lambda'_f + \lambda'_A y_{\max} + \epsilon_k}{\nu \beta_{k-1}} \\ &\leq \frac{2\lambda'_f + 2\lambda'_A y_{\max}}{\nu \beta_{k-1}},\end{aligned} \tag{45}$$

where the second line 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. Let us now revisit and simplify (41). Note that ρ' automatically satisfies the second inequality there, owing to Step 3 of Algorithm 1. Also, ρ satisfies the first inequality in (41) if

$$\frac{\lambda'_f + \lambda'_A y_{\max}}{\nu_A \beta_1} \leq \rho/2, \tag{46}$$

and k_0 is large enough. Indeed, this claim follows directly from (45).

It remains to control the first term in (12). To that end, after recalling Step 2 of Algorithm 1 and applying the triangle inequality, we can write that

$$\begin{aligned}&\operatorname{dist}(-\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) \\ &\leq \operatorname{dist}(-\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_{k-1}), \partial g(x_{k-1})) \\ &+ \|\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k) - \nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_{k-1})\|.\end{aligned} \tag{47}$$

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 (47), we write that

$$\begin{aligned}&\|\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k) - \nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_{k-1})\| \\ &= \|DA(x_k)^\top (y_k - y_{k-1})\| \qquad \text{(see (7))} \\ &\leq \lambda'_A \|y_k - y_{k-1}\| \qquad \text{(see (19))} \\ &= \lambda'_A \sigma_k \|A(x_k)\| \qquad \text{(see Step 5 of Algorithm 1)} \\ &\leq \frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}). \qquad \text{(see (45))}\end{aligned} \tag{48}$$

By combining (47,48), we find that

$$\begin{aligned}&\operatorname{dist}(\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) \\ &\leq \frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k.\end{aligned} \tag{49}$$

By combining (45,49), we find that

$$\begin{aligned}&\operatorname{dist}(-\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) + \|A(x_k)\| \\ &\leq \left(\frac{2\lambda'_A \sigma_k}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k \right) \\ &\quad + 2 \left(\frac{\lambda'_f + \lambda'_A y_{\max}}{\nu \beta_{k-1}} \right).\end{aligned} \tag{50}$$

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

$$\begin{aligned}&\operatorname{dist}(-\nabla_x \mathcal{L}_{\beta_{k-1}}(x_k, y_k), \partial g(x_k)) + \|A(x_k)\| \\ &\leq \frac{2\lambda'_A \sigma_1 + 2}{\nu \beta_{k-1}} (\lambda'_f + \lambda'_A y_{\max}) + \epsilon_k.\end{aligned} \tag{51}$$

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

$$\lambda_{\min}(\nabla_{xx}\mathcal{L}_{\beta_{k-1}}(x_k, y_{k-1})) \geq \lambda_{\min}(\nabla_{xx}\mathcal{L}_{\beta_{k-1}}(x_k, y_k)) - \sigma_k \|\sum_{i=1}^{m} A_i(x_k)\nabla^2 A_i(x_k)\|. \tag{52}$$

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 (5,45). Plugging into (52) gives

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

which completes the proof of Theorem 4.1.

B. Proof of Corollary 4.2

Let K denote the number of (outer) iterations of Algorithm 1 and let ϵ_f denote the desired accuracy of Algorithm 1, see (11). Recalling Theorem 4.1, we can then write that

$$\epsilon_f = \frac{Q}{\beta_K}, \tag{53}$$

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 (17) and for sufficiently large k, recall that $\lambda_{\beta_k} \leq \lambda'' \beta_k$ is the smoothness parameter of the augmented Lagrangian. Then, from (25) ad by summing over the outer iterations, we bound the total number of (inner) iterations of Algorithm 1 as

$$\begin{aligned}
T &= \sum_{k=1}^{K} \mathcal{O}\left(\frac{\lambda^2_{\beta_{k-1}} x^2_{\max}}{\epsilon_k}\right) \\
&= \sum_{k=1}^{K} \mathcal{O}\left(\beta^3_{k-1} x^2_{\max}\right) \qquad \text{(Step 1 of Algorithm 1)} \\
&\leq \mathcal{O}\left(K \beta^3_{K-1} x^2_{\max}\right) \qquad (\{\beta_k\}_k \text{ is increasing}) \\
&\leq \mathcal{O}\left(\frac{K Q^3 x^2_{\max}}{\epsilon^3_f}\right). \qquad \text{(see (53))}
\end{aligned} \tag{54}$$

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

$$\beta_K = b^K \implies K = \log_b\left(\frac{Q}{\epsilon_f}\right), \tag{55}$$

which, after substituting into (54) gives the final bound in Corollary 4.2.

C. Proof of Lemma 2.1

Note that

$$\mathcal{L}_\beta(x, y) = f(x) + \sum_{i=1}^{m} y_i A_i(x) + \frac{\beta}{2}\sum_{i=1}^{m}(A_i(x))^2, \tag{56}$$

which implies that

$$\begin{aligned}
&\nabla_x \mathcal{L}_\beta(x,y) \\
&= \nabla f(x) + \sum_{i=1}^m y_i \nabla A_i(x) + \frac{\beta}{2} \sum_{i=1}^m A_i(x) \nabla A_i(x) \\
&= \nabla f(x) + DA(x)^\top y + \beta DA(x)^\top A(x),
\end{aligned} \tag{57}$$

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

$$\begin{aligned}
\nabla_x^2 \mathcal{L}_\beta(x,y) = \nabla^2 f(x) &+ \sum_{i=1}^m \left(y_i + \beta A_i(x)\right) \nabla^2 A_i(x) \\
&+ \beta \sum_{i=1}^m \nabla A_i(x) \nabla A_i(x)^\top.
\end{aligned} \tag{58}$$

It follows that

$$\begin{aligned}
&\|\nabla_x^2 \mathcal{L}_\beta(x,y)\| \\
&\le \|\nabla^2 f(x)\| + \max_i \|\nabla^2 A_i(x)\| \left(\|y\|_1 + \beta \|A(x)\|_1\right) \\
&\quad + \beta \sum_{i=1}^m \|\nabla A_i(x)\|^2 \\
&\le \lambda_h + \sqrt{m}\lambda_A \left(\|y\| + \beta \|A(x)\|\right) + \beta \|DA(x)\|_F^2.
\end{aligned} \tag{59}$$

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

$$\begin{aligned}
\|\nabla_x^2 \mathcal{L}_\beta(x,y)\| \le \lambda_f + \sqrt{m}\lambda_A \left(\|y\| + \beta\rho\right) \\
+ \beta \max_{\|x\| \le \rho'} \|DA(x)\|_F^2,
\end{aligned} \tag{60}$$

which completes the proof of Lemma 2.1.

D. Basis Pursuit

We only verify the regularity condition in (20) for (1) with f, A specified in (36). Note that

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

where $\text{diag}(x) \in \mathbb{R}^{2d \times 2d}$ is the diagonal matrix formed by x. The left-hand side of (20) then reads as

$$\begin{aligned}
&\text{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right) \\
&= \text{dist}\left(-DA(x_k)^\top A(x_k), \{0\}\right) \\
&= \|DA(x_k)^\top A(x_k)\| \\
&= 2\|\text{diag}(x_k)\overline{B}^\top (\overline{B} x_k^{\circ 2} - b)\|.
\end{aligned} \tag{61}$$

To bound the last line above, let x_* be a solution of Program (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 amplitudes of z_k. To simplify matters, let us assume also that B is full-rank. We then

rewrite the last line of (61) as

$$\begin{aligned}
&\|\text{diag}(x_k)\overline{B}^\top(\overline{B}x_k^{\circ 2} - b)\|^2 \\
&= \|\text{diag}(x_k)\overline{B}^\top\overline{B}(x_k^{\circ 2} - x_*^{\circ 2})\|^2 \\
&= \|\text{diag}(x_k)\overline{B}^\top B(x_k - x_*)\|^2 \\
&= \|\text{diag}(u_{k,1})B^\top B(x_k - x_*)\|^2 \\
&\quad + \|\text{diag}(u_{k,2})B^\top B(x_k - x_*)\|^2 \\
&= \|\text{diag}(u_{k,1}^{\circ 2} + u_{k,2}^{\circ 2})B^\top B(x_k - x_*)\|^2 \\
&= \|\text{diag}(|z_k|)B^\top B(x_k - x_*)\|^2 \\
&\geq \eta_n(B\text{diag}(|z_k|))^2\|B(x_k - x_*)\|^2 \\
&= \eta_n(B\text{diag}(|z_k|))^2\|Bx_k - b\|^2,
\end{aligned} \tag{62}$$

where $\eta_n(\cdot)$ returns the nth largest singular value of its argument. We can therefore ensure that (20) holds by enforcing that

$$z_k \in \left\{z \in \mathbb{R}^d : \eta_n(B\text{diag}(|z|)) > \nu\right\}, \tag{63}$$

for every iteration k.

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 (Obozinski et al., 2011), 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.

E. Clustering

We only verify the condition in (20). Note that

$$A(x) = VV^\top\mathbf{1} - \mathbf{1}, \tag{64}$$

$$\begin{aligned}
DA(x) &= \begin{bmatrix} w_{1,1}x_1^\top & \cdots & w_{1,n}x_1^\top \\ \vdots & & \\ w_{n,1}x_n^\top & \cdots & w_{n,n}1x_n^\top \end{bmatrix} \\
&= \begin{bmatrix} V & \cdots & V \end{bmatrix} + \begin{bmatrix} x_1^\top & & \\ & \ddots & \\ & & x_n^\top \end{bmatrix},
\end{aligned} \tag{65}$$

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 as in the example and let $x_{k,i}$ be the ith row of V_k. Consequently,

$$\begin{aligned}
DA(x_k)^\top A(x_k) &= \begin{bmatrix} V_k^\top(V_kV_k^\top - I_n)\mathbf{1} \\ \vdots \\ V_k^\top(V_kV_k^\top - I_n)\mathbf{1} \end{bmatrix} \\
&\quad + \begin{bmatrix} x_{k,1}(V_kV_k^\top\mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_{k,n}(V_kV_k^\top\mathbf{1} - \mathbf{1})_n \end{bmatrix},
\end{aligned} \tag{66}$$

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 $x_k \in \text{relint}(C)$ so that $\partial g(x_k) = \{0\}$, which can be easily enforced in the iterates. Second, we assume that V_k has nearly orthonormal columns, namely, $V_k V_k^\top \approx I_n$. This can also be easily 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 derivations. Under these assumptions, the squared right-hand side of (20) becomes

$$\begin{aligned}
&\text{dist}\left(-DA(x_k)^\top A(x_k), \frac{\partial g(x_k)}{\beta_{k-1}}\right)^2 \\
&= \text{dist}\left(-DA(x_k)^\top A(x_k), \{0\}\right)^2 \\
&= \|DA(x_k)^\top A(x_k)\|^2 \\
&= \left\| \begin{array}{c} x_{k,1}(V_k V_k^\top \mathbf{1} - \mathbf{1})_1 \\ \vdots \\ x_{k,n}(V_k V_k^\top \mathbf{1} - \mathbf{1})_n \end{array} \right\|^2 \\
&= \sum_{i=1}^n \|x_{k,i}\|^2 (V_k V_k^\top \mathbf{1} - \mathbf{1})_i^2 \\
&\geq \min_i \|x_{k,i}\|^2 \cdot \sum_{i=1}^n (V_k V_k^\top \mathbf{1} - \mathbf{1})_i^2 \\
&= \min_i \|x_{k,i}\|^2 \cdot \|V_k V_k^\top \mathbf{1} - \mathbf{1}\|^2.
\end{aligned} \tag{67}$$

We can enforce the iterates to satisfy $\|x_{k,i}\| \geq \nu$, which corresponds again to well-separated clusters, and guarantee (20). In practice, often n exceeds the number of true clusters and a more fine-tuned analysis is required to establish (20) by restricting the argument to a particular subspace of $\mathbb{R}^n$.

F. ℓ_∞ denoising with a generative prior

The authors of (Ilyas et al., 2017) have proposed to project onto the range of a Generative Adversarial network (GAN) (Goodfellow et al., 2014), 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:

$$\begin{array}{ll} \min\limits_{x,z} & \|x^* + \eta - x\|_\infty \\ \text{s.t.} & x = G(z). \end{array} \tag{68}$$

We use a pretrained generator for the MNIST dataset, given by a standard deconvolutional neural network architecture. We compare the succesful optimizer Adam against our method. Our algorithm involves two forward/backward passes through the network, as oposed to Adam that requires only one. For this reason we let our algorithm run for 4000 iterations, and Adam for 8000 iterations. For a particular example, we plot the objective value vs iteration count in figure F. Our method successfully minimizes the objective value, while Adam does not succeed.

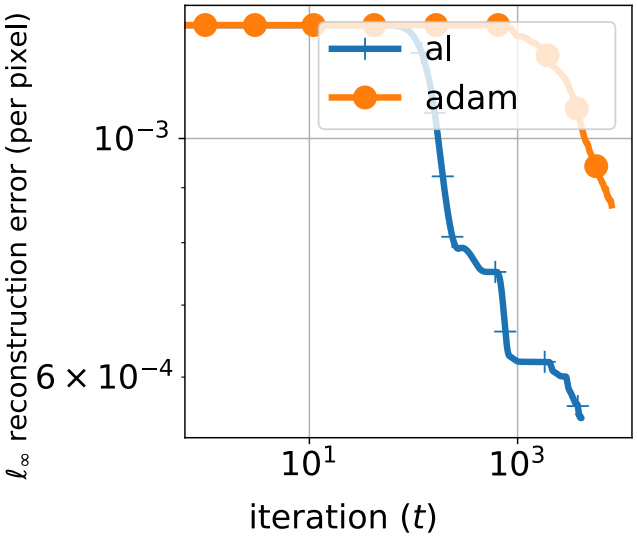

Figure 4. Augmented Lagrangian vs Adam for ℓ_∞ denoising (left). ℓ_2 vs ℓ_∞ denoising as defense against adversarial examples