Adaptive Relaxed ADMM: Convergence Theory and Practical Implementation

Zheng Xu[1]*, Mário A. T. Figueiredo[2], Xiaoming Yuan[3], Christoph Studer[4], and Tom Goldstein[1]
[1]Department of Computer Science, University of Maryland, College Park, MD
[2]Instituto de Telecomunicações, Instituto Superior Técnico, Universidade de Lisboa, Portugal
[3]Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong
[4]School of Electrical and Computer Engineering, Cornell University, Ithaca, NY

Abstract

Many modern computer vision and machine learning applications rely on solving difficult optimization problems that involve non-differentiable objective functions and constraints. The alternating direction method of multipliers (ADMM) is a widely used approach to solve such problems. Relaxed ADMM is a generalization of ADMM that often achieves better performance, but its efficiency depends strongly on algorithm parameters that must be chosen by an expert user. We propose an adaptive method that automatically tunes the key algorithm parameters to achieve optimal performance without user oversight. Inspired by recent work on adaptivity, the proposed adaptive relaxed ADMM *(ARADMM) is derived by assuming a Barzilai-Borwein style linear gradient. A detailed convergence analysis of ARADMM is provided, and numerical results on several applications demonstrate fast practical convergence.*

1. Introduction

Modern methods in computer vision and machine learning often require solving difficult optimization problems involving non-differentiable objective functions and constraints. Some popular applications include sparse models [48, 54, 8, 36], low-rank models [47, 23, 53, 31], and support vector machines (SVMs) [4, 3]. The alternating direction method of multiplier (ADMM) is one of the most prominent optimization tools to solve such problems, and tackles problems in the following form:

$$\min_{u\in\mathbb{R}^n, v\in\mathbb{R}^m} \; h(u)+g(v), \quad \text{subject to } \; Au+Bv=b. \tag{1}$$

Here, $h:\mathbb{R}^n\to\mathbb{R}$ and $g:\mathbb{R}^m\to\mathbb{R}$ are closed, proper, and convex functions, $A\in\mathbb{R}^{p\times n}$, $B\in\mathbb{R}^{p\times m}$, and $b\in\mathbb{R}^p$. ADMM was first introduced in [16] and [12], and has found applications in a variety of optimization problems in machine learning, image processing, computer vision, wireless communications, and many other areas [2, 21].

Relaxed ADMM is a popular practical variant of ADMM, and proceeds with the following steps:

$$u_{k+1} = \arg\min_u h(u) + \frac{\tau_k}{2}\left\| b - Au - Bv_k + \frac{\lambda_k}{\tau_k}\right\|^2 \tag{2}$$

$$\tilde{u}_{k+1} = \gamma_k A u_{k+1} + (1-\gamma_k)(b - Bv_k) \tag{3}$$

$$v_{k+1} = \arg\min_v g(v) + \frac{\tau_k}{2}\left\| b - \tilde{u}_{k+1} - Bv + \frac{\lambda_k}{\tau_k}\right\|^2 \tag{4}$$

$$\lambda_{k+1} = \lambda_k + \tau_k(b - \tilde{u}_{k+1} - Bv_{k+1}). \tag{5}$$

Here, $\lambda_k\in\mathbb{R}^p$ denotes the dual variables (Lagrange multipliers) on iteration k, and (τ_k,γ_k) are sequences of penalty and relaxation parameters. Relaxed ADMM coincides with the original non-relaxed version if $\gamma_k=1$.

Convergence of (relaxed) ADMM is guaranteed under fairly general assumptions [6, 25, 26, 10], if the penalty and relaxation parameters are held constant. However, the practical performance of ADMM depends strongly on the choice of these parameters, as well as on the problem being solved. Good penalty choices are known for certain ADMM formulations, such as strictly convex quadratic problems [40, 14], and for the gradient descent parameter in the "linearized" ADMM [32, 34].

Adaptive penalty methods (in which the penalty parameters are tuned automatically as the algorithm proceeds) achieve good performance without user oversight. For non-relaxed ADMM, the authors of [24] propose methods that modulate the penalty parameter so that the primal and dual residuals (i.e., derivatives of the Lagrangian with respect to primal and dual variables) are of approximately equal size. This "residual balancing" approach has been generalized to work with preconditioned variants of ADMM [20] and distributed ADMM [44]. In [51], a spectral penalty parameter method is proposed that uses the local curvature of the objective to achieve fast convergence. All of these methods are

*xuzh@cs.umd.edu

specific to (non-relaxed) *vanilla* ADMM, and do not apply to the more general case involving a relaxation parameter.

1.1. Overview & contributions

In this paper, we study adaptive parameter choices for the relaxed ADMM that jointly and automatically tune both the penalty parameter τ_k and relaxation parameter γ_k. In Section 3, we address theoretical questions about the convergence of ADMM with non-constant penalty and relaxation parameters. In Section 4, we discuss practical methods for choosing these parameters. In Section 6, we apply the proposed ARADMM to several problems in machine learning, computer vision, and image processing. Finally, in Section 7, we compare ARADMM to other ADMM variants and examine the benefits of the proposed approach for real-world regression, classification, and image processing problems.

2. Related work

Sparse and low rank methods are widely used in computer vision [48, 54, 8, 47, 23, 36, 53, 31], machine learning [7, 57, 43, 9, 33], and image processing [42, 21]. ADMM has been extensively applied to solve such problems [2, 21, 51, 50], and has recently found applications in neural networks [56, 45], tensor decomposition [18, 35, 52], structure from motion [19], and other vision problems.

The $O(1/k)$ convergence rate of non-relaxed ADMM is established under mild conditions for convex problems [25, 26]. The $O(1/k^2)$ convergence rate is discussed in [17, 21, 27, 46], where at least one of the functions is assumed either strongly convex or smooth. For the general relaxed ADMM formulation, a $O(1/k)$ convergence rate is provided under mild conditions [10]. Linear convergence can be achieved with strong convexity assumptions [5, 38, 15]. All of these results assume constant parameters—it is considerably harder to prove convergence when the algorithm parameters are adaptive.

Fixed optimal parameters are discussed in the literature. For the specific case in which the objective is quadratic, a criterion is proposed in [40, 14]. The authors of [38] suggest a grid search and semidefinite programming based method to determine the optimal relaxation and penalty parameters. These methods, however, make strong assumptions about the objective and require knowledge of condition numbers.

Adaptive penalty methods are proposed to accelerate the practical convergence of non-relaxed ADMM [24, 51]. For the relaxation parameter, it has been suggested in [6] that over-relaxation ($\gamma \in (1, 2)$) may accelerate convergence and $\gamma = 1.5$ achieves faster convergence in a specific distributed computing application. The proposed ARADMM simultaneously adapts both the penalty and the relaxation parameter, thus being fully automated.

3. Convergence theory

We study conditions under which ADMM converges with adaptive penalty and relaxation parameters. Our approach utilizes the variational inequality (VI) methods put forward in [24, 25, 26]. Our results measure convergence using the primal and dual "residuals," which are defined as

$$r_k = b - Au_k - Bv_k \text{ and } d_k = \tau_k A^T B(v_k - v_{k-1}). \quad (6)$$

It has been observed that these residuals approach zero as the algorithm approaches a true solution [2]. Typically, the iterative process is stopped if

$$\begin{aligned} &\|r_k\| \leq \epsilon^{tol} \max\{\|Au_k\|, \|Bv_k\|, \|b\|\} \\ &\text{and } \|d_k\| \leq \epsilon^{tol} \|A^T \lambda_k\|, \end{aligned} \quad (7)$$

where $\epsilon^{tol} > 0$ is the stopping tolerance [2]. For this reason, it is important to know that the method converges in the sense that the residuals approach zero as $k \to \infty$.

In the sequel, we prove that relaxed ADMM converges in the residual sense, provided that the algorithm parameters satisfy one of the following two assumptions.

Assumption 1. *The relaxation sequence γ_k and penalty sequence τ_k satisfy*

$$\begin{aligned} &1 \leq \gamma_k < 2, \ \lim_{k\to\infty} 1/\tau_k^2 < \infty, \ \sum_{k=1}^{\infty} \eta_k^2 < \infty, \\ &\textit{where} \quad \eta_k^2 = \frac{\gamma_k}{(2-\gamma_k)} \max\left(\tau_k^2/\tau_{k-1}^2, 1\right) - 1. \end{aligned} \quad (8)$$

Assumption 2. *The relaxation sequence γ_k and penalty sequence τ_k satisfy*

$$\begin{aligned} &1 \leq \gamma_k < 2, \ \lim_{k\to\infty} \tau_k^2 < \infty, \ \sum_{k=1}^{\infty} \theta_k^2 < \infty, \\ &\textit{where} \quad \theta_k^2 = \frac{\gamma_k}{(2-\gamma_k)} \max\left(\tau_{k-1}^2/\tau_k^2, 1\right) - 1. \end{aligned} \quad (9)$$

In Section 5, we prove adaptive relaxed ADMM converges if the algorithm parameters satisfy either Assumption 1 or Assumption 2. Before presenting the proof, we show how to choose the relaxation parameters that lead to efficient performance in practice.

4. ARADMM: Adaptive relaxed ADMM

Spectral stepsize selection methods for vanilla ADMM were discussed in [51]. Here, we modify the adaptive ADMM framework in two important ways. First, we discuss the selection of penalty parameters in the presence of the relaxation term. Second, we discuss adaptive methods also for automatically selecting the relaxation parameter.

The proposed method works by assuming a local linear model for the dual optimization problem, and then selecting

an optimal stepsize under this assumption. A safeguarding method is adopted to ensure that bad stepsizes are not chosen in case these linearity assumptions fail to hold.

4.1. Dual interpretation of relaxed ADMM

We derive our adaptive stepsize rules by examining the close relationship between relaxed ADMM and the relaxed Douglas-Rachford Splitting (DRS) [6, 5, 15]. The dual of the general constrained problem (1) is

$$\min_{\zeta\in\mathbb{R}^p} \underbrace{h^*(A^T\zeta) - \langle \zeta, b\rangle}_{\hat{h}(\zeta)} + \underbrace{g^*(B^T\zeta)}_{\hat{g}(\zeta)}, \tag{10}$$

with f^* denoting the Fenchel conjugate of f, defined as $f^*(y) = \sup_x \langle x, y\rangle - f(x)$ [41].

The relaxed DRS algorithm solves (10) by generating two sequences, $(\zeta_k)_{k\in\mathbb{N}}$ and $(\hat{\zeta}_k)_{k\in\mathbb{N}}$, according to

$$0 \in \frac{\hat{\zeta}_{k+1} - \zeta_k}{\tau_k} + \partial\hat{h}(\hat{\zeta}_{k+1}) + \partial\hat{g}(\zeta_k), \tag{11}$$

$$\begin{aligned} 0 \in &\frac{\zeta_{k+1} - \zeta_k}{\tau_k} + \gamma_k\,\partial\hat{h}(\hat{\zeta}_{k+1}) \\ &- (1-\gamma_k)\partial\hat{g}(\zeta_k) + \partial\hat{g}(\zeta_{k+1}), \end{aligned} \tag{12}$$

where γ_k is a relaxation parameter, and $\partial f(x)$ denotes the subdifferential of f evaluated at x [41]. Referring back to ADMM in (2)–(5), and defining $\hat{\lambda}_{k+1} = \lambda_k + \tau_k(b - Au_{k+1} - Bv_k)$, the sequences $(\lambda_k)_{k\in\mathbb{N}}$ and $(\hat{\lambda}_k)_{k\in\mathbb{N}}$ satisfy the same conditions (11) and (12) as $(\zeta_k)_{k\in\mathbb{N}}$ and $(\hat{\zeta}_k)_{k\in\mathbb{N}}$, thus ADMM for the problem (1) is equivalent to DRS on its dual (10). A detailed proof of this is provided in the supplementary material.

4.2. Spectral adaptive stepsize rule

Adaptive stepsize rules of the "spectral" type were originally proposed for simple gradient descent on smooth problems by Barzilai and Borwein [1], and have been found to dramatically outperform constant stepsizes in many applications [11, 49]. Spectral stepsize methods work by modeling the gradient of the objective as a linear function, and then selecting the optimal stepsize for this simplified linear model.

Spectral methods were recently used to determine the penalty parameter for the non-relaxed ADMM in [51]. Inspired by that work, we derive spectral stepsize rules assuming a linear model/approximation for $\partial\hat{h}(\hat{\zeta})$ and $\partial\hat{g}(\zeta)$ at iteration k given by

$$\partial\hat{h}(\hat{\zeta}) = \alpha_k\,\hat{\zeta} + \Psi_k \quad \text{and} \quad \partial\hat{g}(\zeta) = \beta_k\,\zeta + \Phi_k, \tag{13}$$

where $\alpha_k > 0$, $\beta_k > 0$ are local curvature estimates of $\hat{h}$ and $\hat{g}$, respectively, and $\Psi_k, \Phi_k \subset \mathbb{R}^p$. Once we obtain these curvature estimates, we will exploit the following simple proposition whose proof is given in the supplementary material.

Proposition 1. *Suppose the DRS steps (11)–(12) are applied to problem (10), where (omitting iteration k from $\alpha_k, \beta_k, \Psi_k, \Phi_k$ to lighten the notation in what follows)*

$$\partial\hat{h}(\hat{\zeta}) = \alpha\,\hat{\zeta} + \Psi \quad \text{and} \quad \partial\hat{g}(\zeta) = \beta\,\zeta + \Phi. \tag{14}$$

Then, the residual of $\hat{h}(\zeta_{k+1}) + \hat{g}(\zeta_{k+1})$ will be zero if τ and γ are chosen to satisfy $\gamma_k = 1 + \frac{1+\alpha\beta\tau_k^2}{(\alpha+\beta)\tau_k}$.

Our adaptive method works by fitting a linear model to the gradient (or subgradient) of our objective, and then using Proposition 1 to select an optimal stepsize pair that obtains zero residual on the model problem. For our convergence theory to hold, we need $\gamma < 2$. For fixed values of α and β, the minimal value of γ_k that is still optimal for the linear model occurs if we choose

$$\tau_k = \arg\min_{\tau} \frac{1+\alpha\beta\tau^2}{(\alpha+\beta)\tau} = 1/\sqrt{\alpha\beta}. \tag{15}$$

Note this is the same "optimal" penalty parameter proposed for non-relaxed ADMM in [51]. Under this choice of τ_k, we then have the "optimal" relaxation parameter

$$\gamma_k = 1 + \frac{1+\alpha\beta\tau^2}{(\alpha+\beta)\tau} = 1 + \frac{2\sqrt{\alpha\beta}}{\alpha+\beta} \le 2. \tag{16}$$

4.3. Estimation of stepsizes

We now propose a simple method for fitting a linear model to the dual objective terms so that the formulas in Section 4.2 can be used to obtain stepsizes. Once these linear models are formed, the optimal penalty parameter and relaxation term can be calculated by (15) and (16), thanks to the equivalence of relaxed ADMM and DRS.

In what follows, we let $\hat{\alpha}_k = 1/\alpha_k$ and $\hat{\beta}_k = 1/\beta_k$ to simplify notation. The optimal stepsize choice is then written as $\tau_k = (\hat{\alpha}_k\,\hat{\beta}_k)^{1/2}$ and $\gamma_k = 1 + \frac{2\sqrt{\hat{\alpha}_k\hat{\beta}_k}}{\hat{\alpha}_k+\hat{\beta}_k}$.

The estimation of $\hat{\alpha}_k$ and $\hat{\beta}_k$ for the dual components $\hat{h}(\hat{\lambda}_k)$ and $\hat{g}(\lambda_k)$ at the k-th iteration of primal ADMM has been described in [51]. It is easy to verify that the model parameters $\hat{\alpha}_k$ and $\hat{\beta}_k$ of relaxed ADMM can be estimated based on the results from iteration k and an older iteration $k_0 < k$ in a similar way. If we define

$$\Delta\hat{\lambda}_k := \hat{\lambda}_k - \hat{\lambda}_{k_0} \quad \text{and} \quad \Delta\hat{h}_k := A(u_k - u_{k_0}), \tag{17}$$

then the parameter $\hat{\alpha}_k$ is obtained from the formula

$$\hat{\alpha}_k = \begin{cases} \hat{\alpha}_k^{\text{MG}} & \text{if } 2\,\hat{\alpha}_k^{\text{MG}} > \hat{\alpha}_k^{\text{SD}} \\ \hat{\alpha}_k^{\text{SD}} - \hat{\alpha}_k^{\text{MG}}/2 & \text{otherwise,} \end{cases} \tag{18}$$

$$\hat{\alpha}_k^{\text{SD}} = \frac{\langle \Delta\hat{\lambda}_k, \Delta\hat{\lambda}_k\rangle}{\langle \Delta\hat{h}_k, \Delta\hat{\lambda}_k\rangle} \text{ and } \hat{\alpha}_k^{\text{MG}} = \frac{\langle \Delta\hat{h}_k, \Delta\hat{\lambda}_k\rangle}{\langle \Delta\hat{h}_k, \Delta\hat{h}_k\rangle}. \tag{19}$$

For a detailed derivation of these formulas, see [51].

The spectral stepsize $\hat{\beta}_k$ of $\hat{g}(\lambda_k)$ is similarly estimated with $\Delta\hat{g}_k := B(v_k - v_{k_0})$, and $\Delta\lambda_k := \lambda_k - \lambda_{k_0}$. It is important to note that $\hat{\alpha}_k$ and $\hat{\beta}_k$ are obtained from the iterates of ADMM alone, i.e., our scheme does not require the user to supply the dual problem.

4.4. Safeguarding

Spectral stepsize methods for simple gradient descent are paired with a backtracking line search to guarantee convergence in case the linear model assumptions break down and an unstable stepsize is produced. ADMM methods have no analog of backtracking. Rather, we adopt the correlation criterion proposed in [51] to test the validity of the local linear assumption, and only rely on the adaptive model when the assumptions are deemed valid. To this end, we define

$$\alpha_k^{\text{cor}} = \frac{\langle \Delta\hat{h}_k, \Delta\hat{\lambda}_k \rangle}{\|\Delta\hat{h}_k\| \, \|\Delta\hat{\lambda}_k\|} \quad \text{and} \quad \beta_k^{\text{cor}} = \frac{\langle \Delta\hat{g}_k, \Delta\lambda_k \rangle}{\|\Delta\hat{g}_k\| \, \|\Delta\lambda_k\|}. \tag{20}$$

When the model assumptions (14) hold perfectly, the vectors $\Delta\hat{h}_k$ and $\Delta\hat{\lambda}_k$ should be highly correlated and we get $\alpha_k^{\text{cor}} = 1$. When α_k^{cor} or β_k^{cor} is small, the model assumptions are invalid and the spectral stepsize may not be effective.

The proposed method uses the following update rules

$$\tau_{k+1} = \begin{cases} \sqrt{\hat{\alpha}_k \hat{\beta}_k} & \text{if } \alpha_k^{\text{cor}} > \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} > \epsilon^{\text{cor}} \\ \hat{\alpha}_k & \text{if } \alpha_k^{\text{cor}} > \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} \le \epsilon^{\text{cor}} \\ \hat{\beta}_k & \text{if } \alpha_k^{\text{cor}} \le \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} > \epsilon^{\text{cor}} \\ \tau_k & \text{otherwise,} \end{cases} \tag{21}$$

$$\gamma_{k+1} = \begin{cases} 1 + \frac{2\sqrt{\hat{\alpha}_k \hat{\beta}_k}}{\hat{\alpha}_k + \hat{\beta}_k} & \text{if } \alpha_k^{\text{cor}} > \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} > \epsilon^{\text{cor}} \\ 1.9 & \text{if } \alpha_k^{\text{cor}} > \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} \le \epsilon^{\text{cor}} \\ 1.1 & \text{if } \alpha_k^{\text{cor}} \le \epsilon^{\text{cor}} \text{ and } \beta_k^{\text{cor}} > \epsilon^{\text{cor}} \\ 1.5 & \text{otherwise,} \end{cases} \tag{22}$$

where ϵ^{cor} is a quality threshold for the curvature estimates, while $\hat{\alpha}_k$ and $\hat{\beta}_k$ are the spectral stepsizes estimated in Section 4.3. The update for τ_{k+1} only uses model parameters that have been accurately estimated. When the model is effective for h but not g, we use a large $\gamma_k = 1.9$ to make the v update conservative relative to the u update. When the model is effective for g but not h, we use a small $\gamma_k = 1.1$ to make the v update aggressive relative to the u update.

4.5. Applying convergence guarantee

Our convergence theory requires either Assumption 1 or Assumption 2 to be satisfied, which suggests that convergence is guaranteed under "bounded adaptivity" for both penalty and relaxation parameters. These conditions can be guaranteed by explicitly adding constraints to the stepsize choice in ARADMM.

Algorithm 1 Adaptive relaxed ADMM (ARADMM)

Input: initialize v_0, λ_0, τ_0, γ_0, and $k_0 = 0$

while not converge by (7) **and** $k <$ maxiter **do**
  Perform relaxed ADMM, as in (2)–(5)
  if mod$(k, T_f) = 1$ **then**
    $\hat{\lambda}_{k+1} = \lambda_k + \tau_k(b - Au_{k+1} - Bv_k)$
    Compute spectral stepsizes $\hat{\alpha}_k, \hat{\beta}_k$ using (18)
    Estimate correlations $\alpha_k^{\text{cor}}, \beta_k^{\text{cor}}$ using (20)
    Update τ_{k+1}, γ_{k+1} using (21) and (22)
    Bound τ_{k+1}, γ_{k+1} using (23)
    $k_0 \leftarrow k$
  else
    $\tau_{k+1} \leftarrow \tau_k$ and $\gamma_{k+1} \leftarrow \gamma_k$
  end if
  $k \leftarrow k + 1$
end while

To guarantee convergence, we simply replace the parameter updates (21) and (22) with

$$\begin{aligned} \hat{\tau}_{k+1} &= \min\left\{\tau_{k+1},\ (1 + {}^{C_{cg}}/_{k^2})\,\tau_k\right\} \\ \hat{\gamma}_{k+1} &= \min\left\{\gamma_{k+1},\ 1 + {}^{C_{cg}}/_{k^2}\right\}, \end{aligned} \tag{23}$$

where C_{cg} is some (large) constant. It is easily verified that the parameter sequence $(\hat{\tau}_k, \hat{\gamma}_k)$ satisfies Assumption 1. In practice, the update schemes (21) and (22) converges reliably without explicitly enforcing these conditions. We use a very large C_{cg} such that the conditions are not triggered in the first few thousand iterations and provide these constraints for theoretical interests.

4.6. ARADMM algorithm

The complete *adaptive relaxed ADMM* (ARADMM) is shown in Algorithm 1. We suggest only updating the stepsize every $T_f = 2$ iterations. We suggest a fixed safeguarding threshold $\epsilon^{\text{cor}} = 0.2$, which is used in all the experiments in Section 6. The overhead of the adaptive scheme is modest, requiring only a few inner product calculations.

5. Proofs of convergence theorems

We now prove that relaxed ADMM converges under Assumption 1 or 2. Let

$$y = \begin{pmatrix} u \\ v \end{pmatrix} \in \mathbb{R}^{n+m}, \quad z = \begin{pmatrix} u \\ v \\ \lambda \end{pmatrix} \in \mathbb{R}^{n+m+p}. \tag{24}$$

We use $y_k = (u_k, v_k)^T$ and $z_k = (u_k, v_k, \lambda_k)^T$ to denote iterates, and $y^* = (u^*, v^*)^T$ and $z^* = (u^*, v^*, \lambda^*)^T$ denote optimal solutions. Set $\Delta z_k^+ = (\Delta u_k^+, \Delta v_k^+, \Delta\lambda_k^+) := z_{k+1} - z_k$, and $\Delta z_k^* = (\Delta u_k^*, \Delta v_k^*, \Delta\lambda_k^*) := z^* - z_k$, and

define

$$f(y) = h(u) + g(v), \quad F(z) = \begin{pmatrix} -A^T\lambda \\ -B^T\lambda \\ Au + Bv - b \end{pmatrix}. \quad (25)$$

Notice that $F(z)$ is monotone, which means $\forall z, z', (z - z')^T(F(z) - F(z')) \geq 0$.

Problem formulation (1) can be reformulated as a variational inequality (VI). The optimal solution z^* satisfies

$$\forall z, \;\; f(y) - f(y^*) + (z - z^*)^T F(z^*) \geq 0. \quad (26)$$

Likewise, the ADMM iterates produced by steps (2) and (4) satisfy the variational inequalities

$$\forall u, \; h(u) - h(u_{k+1}) + (u - u_{k+1})^T (\tau_k A^T(Au_{k+1} + Bv_k - b) - A^T\lambda_k) \geq 0, \quad (27)$$

$$\forall v, \; g(v) - g(v_{k+1}) + (v - v_{k+1})^T (\tau_k B^T(\tilde{u}_{k+1} + Bv_{k+1} - b) - B^T\lambda_k) \geq 0. \quad (28)$$

Using the definitions of y, z, $f(y)$, and $F(z)$ in (24, 25), λ in (5), and $\tilde{u}$ in (3), VI (27) and (28) combine to yield

$$f(y) - f(y_{k+1}) + (z - z_{k+1})^T \left(F(z_{k+1}) + \Omega(\Delta z_k^+, \tau_k, \gamma_k)\right) \geq 0,$$

$$\Omega(\Delta z_k^+, \tau_k, \gamma_k) = \begin{pmatrix} \frac{\gamma_k - 1}{\gamma_k} A^T \Delta\lambda_k^+ - \frac{\tau_k}{\gamma_k} A^T B \Delta v_k^+ \\ 0 \\ \frac{1}{\gamma_k \tau_k} \Delta\lambda_k^+ - \frac{\gamma_k - 1}{\gamma_k} B\Delta v_k^+ \end{pmatrix}. \quad (29)$$

We then apply VI (26), (28), and (29) in order to prove the following lemmas for our contraction proof, which show that the difference between iterates decreases as the iterates approach the true solution. 'The remaining details of the proof are in the supplementary material.

Lemma 1. *The iterates $z_k = (u_k, v_k, \lambda_k)^T$ generated by ADMM satisfy*

$$(B\Delta v_k^+)^T \Delta\lambda_k^+ \geq 0. \quad (30)$$

Lemma 2. *Let $\gamma_k \geq 1$. The optimal solution z^* and iterates z_k generated by ADMM satisfy*

$$\begin{aligned} &\frac{2 - \gamma_k}{\gamma_k} \|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 \\ &\quad \leq \gamma_k(\|\tau_k B\Delta v_k^*\|^2 + \|\Delta\lambda_k^*\|^2) \\ &\qquad - (2 - \gamma_k)(\|\tau_k B\Delta v_{k+1}^*\|^2 + \|\Delta\lambda_{k+1}^*\|^2). \end{aligned} \quad (31)$$

5.1. Convergence with adaptivity

We are now ready to state our main convergence results. The proof of Theorem 1 is shown here in full, and leverages Lemma 2 to produce a contraction argument. The proof of Theorem 2 is extremely similar, and is shown in the supplementary material.

Theorem 1. *Suppose Assumption 1 holds. Then, the iterates $z_k = (u_k, v_k, \lambda_k)^T$ generated by ADMM satisfy*

$$\lim_{k\to\infty} \|r_k\| = 0 \quad \textit{and} \quad \lim_{k\to\infty} \|d_k\| = 0. \quad (32)$$

Proof. Assumption 1 implies

$$\frac{\gamma_k}{2 - \gamma_k}\tau_k^2 \leq (1 + \eta_k^2)\tau_{k-1}^2 \text{ and } \frac{\gamma_k}{2 - \gamma_k} \leq (1 + \eta_k^2). \quad (33)$$

If $\gamma_k < 2$ as in Assumption 1, then Lemma 2 shows

$$\begin{aligned} &\frac{1}{\gamma_k}\|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 \\ &\leq \frac{\gamma_k}{2 - \gamma_k}(\tau_k^2\|B\Delta v_k^*\|^2 + \|\Delta\lambda_k^*\|^2) \\ &\quad - (\tau_k^2\|B\Delta v_{k+1}^*\|^2 + \|\Delta\lambda_{k+1}^*\|^2) \quad (34) \\ &\leq (1 + \eta_k^2)(\tau_{k-1}^2\|B\Delta v_k^*\|^2 + \|\Delta\lambda_k^*\|^2) \\ &\quad - (\tau_k^2\|B\Delta v_{k+1}^*\|^2 + \|\Delta\lambda_{k+1}^*\|^2), \quad (35) \end{aligned}$$

where (33) is used to get from (34) to (35). Accumulating inequality (35) from $k = 0$ to N shows

$$\begin{aligned} &\sum_{k=0}^{N} \prod_{t=k+1}^{N} (1 + \eta_t^2)\frac{1}{\gamma_k}\|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 \\ &\quad \leq \prod_{k=1}^{N}(1 + \eta_t^2)(\tau_0^2\|B\Delta v_0^*\|^2 + \|\Delta\lambda_0^*\|^2). \end{aligned} \quad (36)$$

Assumption 1 also implies $\prod_{t=1}^{\infty}(1 + \eta_t^2) < \infty$, and $\prod_{t=k+1}^{N}(1 + \eta_t^2)\frac{1}{\gamma_k} \geq \frac{1}{\gamma_k} > 1/2$. Then, (36) indicates $\sum_{k=0}^{\infty} \|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 < \infty$, and

$$\lim_{k\to\infty} \|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 = 0. \quad (37)$$

Now, from Lemma 1, $(B\Delta v_k^+)^T\Delta\lambda_k^+ \geq 0$, and so

$$\lim_{k\to\infty} \|\Delta\lambda_k^+\|^2 \leq \lim_{k\to\infty} \|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 = 0, \quad (38)$$

$$\lim_{k\to\infty} \|\tau_k B\Delta v_k^+\|^2 \leq \lim_{k\to\infty} \|\tau_k B\Delta v_k^+ + \Delta\lambda_k^+\|^2 = 0. \quad (39)$$

The residuals r_k, d_k in (6) satisfy

$$r_k = \frac{1}{\gamma_k\tau_k}\Delta\lambda_{k-1}^+ - \frac{\gamma_k - 1}{\gamma_k} B\Delta v_{k-1}^+, \quad (40)$$

$$d_k = \tau_k A^T B\Delta v_{k-1}^+, \quad (41)$$

from which we get

$$\begin{aligned} \lim_{k\to\infty} \|r_k\| &\leq \lim_{k\to\infty} \frac{1}{\gamma_k\tau_k}\|\Delta\lambda_{k-1}^+\| \\ &\quad + \frac{\gamma_k - 1}{\gamma_k\tau_{k-1}^2}\|\tau_{k-1}B\Delta v_{k-1}^+\| = 0, \text{ and} \end{aligned}$$

$$\begin{aligned} \lim_{k\to\infty} \|d_k\| &\leq \lim_{k\to\infty} \|A\|\|\tau_k B\Delta v_{k-1}^+\| \\ &\leq \lim_{k\to\infty} \sqrt{1 + \eta_k^2}\|A\| \, \|\tau_{k-1}B\Delta v_{k-1}^+\| = 0. \end{aligned}$$

□

Similar methods can be used to prove the following about convergence under Assumption 2. The proof of the following theorem is given in the supplementary material.

Theorem 2. *Suppose Assumption 2 holds. Then, the iterates* $z_k = (u_k, v_k, \lambda_k)^T$ *generated by ADMM satisfy*

$$\lim_{k\to\infty} \|r_k\| = 0 \quad \textit{and} \quad \lim_{k\to\infty} \|d_k\| = 0. \tag{42}$$

6. Applications

We focus on the following statistical and image processing problems involving non-differentiable objectives: linear regression with elastic net regularization (EN), low-rank least squares (LRLS), quadratic programming (QP), consensus ℓ_1-regularized logistic regression, support vector machine (SVM), total variation image restoration (TVIR), and robust principle component analysis (RPCA). We study several vision benchmark datasets such as the extended Yale B face dataset [13], MNIST digital images [29], and CIFAR10 object images[1] [28]. We also use synthetic and benchmark datasets from [7, 57, 30, 43, 33, 21], which are obtained from the UCI repository and the LIBSVM page. The experimental setups for each problem are briefly described here, and the implementation details are provided in the supplementary material.

Linear regression with EN regularization Elastic net (EN) is a modification of the ℓ_1-norm (or LASSO) regularizer that helps dealing with highly correlated variables [57, 21], and requires solving

$$\min_x \frac{1}{2}\|Dx - c\|_2^2 + \rho_1\|x\|_1 + \frac{\rho_2}{2}\|x\|_2^2, \tag{43}$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm, D is the data matrix, c contains measurements, and x is the vector of regression coefficient.

Low-rank least squares (LRLS) The nuclear norm (the ℓ_1-norm of the matrix singular values) is a convex surrogate for matrix rank. ADMM has been applied to solve low rank least squares problems [55, 53]

$$\min_X \frac{1}{2}\|DX - C\|_F^2 + \rho_1\|X\|_* + \frac{\rho_2}{2}\|X\|_F^2, \tag{44}$$

where $\|\cdot\|_*$ denotes the nuclear norm, $\|\cdot\|_F$ denotes the Frobenius norm, $D \in \mathbb{R}^{n\times m}$ is a data matrix, $C \in \mathbb{R}^{n\times d}$ contains measurements, and $X \in \mathbb{R}^{m\times d}$ contains variables.

ADMM is applied by splitting the regression term and the non-differentiable regularizer composed of nuclear and Frobenius norm. LRLS has been used to formulate exemplar classifiers and discover visual subcategories [53].

[1]We use the first batch of CIFAR10 that contains 10000 samples.

SVM and QP Support vector machine (SVM) is one of the most successful binary classifiers for computer vision. The dual of the SVM is a QP problem,

$$\begin{aligned} \min_z \quad & \frac{1}{2}z^T Q z - e^T z \\ \text{subject to} \quad & c^T z = 0 \text{ and } 0 \le z \le C, \end{aligned}$$

where z is the SVM dual variable, Q is the kernel matrix, c is a vector of labels, e is a vector of ones, and $C > 0$ [3]. The canonical QP is also considered,

$$\min_x \frac{1}{2}x^T Q x + q^T x \quad \text{subject to } \; Dx \le c. \tag{45}$$

Consensus ℓ_1-regularized logistic regression ADMM has become an important tool for solving distributed optimization problems [2]. A typical problem is the consensus ℓ_1-regularized logistic regression

$$\begin{aligned} &\min_{x_i,z} \sum_{i=1}^{N}\sum_{j=1}^{n_i} \log(1+\exp(-c_j D_j x_i)) + \rho\|z\|_1 \\ &\text{subject to } \; x_i - z = 0, i = 1,\dots,N, \end{aligned} \tag{46}$$

where $x_i \in \mathbb{R}^m$ represents the local variable on the ith distributed node, z is the global variable, n_i is the number of samples in the ith block, $D_j \in \mathbb{R}^m$ is the jth sample, and $c_j \in \{-1, +1\}$ is the corresponding label.

Unwrapped SVM The unwrapped formulation of SVM [22], which can be used in distributed computing environments via "transpose reduction" tricks, applies ADMM to the primal form of SVM to solve

$$\min_x \frac{1}{2}\|x\|_2^2 + C\sum_{j=1}^{n}\max\{1 - c_j D_j^T x,\, 0\}, \tag{47}$$

where $D_j \in \mathbb{R}^m$ is the jth sample of training data, and $c_j \in \{-1, 1\}$ is the corresponding label. ADMM is applied by splitting the ℓ_2-norm regularizer and the non-differentiable hinge loss term.

Total variation image denoising (TVID) Total variation image denoising is often performed by solving [42]

$$\min_x \frac{1}{2}\|x - c\|_2^2 + \rho\|\nabla x\|_1 \tag{48}$$

where c represents given noisy image, and ∇ is the discrete gradient operator, which computes differences between adjacent image pixels. ADMM is applied by splitting the ℓ_2-norm term and the non-differentiable total variation term.

RPCA Robust principal component analysis (RPCA) has broad applications in computer vision and imaging [47, 37, 39]. RPCA recovers a low-rank matrix and a sparse matrix by solving

$$\min_{Z,E} \|Z\|_* + \rho\|E\|_1 \;\text{ subject to }\; Z + E = C, \tag{49}$$

Table 1. Iterations (and runtime in seconds) for various applications. Absence of convergence after n iterations is indicated as $n+$.

Application	Dataset	#samples × #features[1]	Vanilla ADMM	Relaxed ADMM	Residual balance	Adaptive ADMM	**Proposed ARADMM**
Elastic net regression	Synthetic	50 × 40	2000+(.642)	2000+(.660)	424(.144)	102(.051)	**70(.026)**
	MNIST	60000 × 784	1225(29.4)	816(19.9)	94(2.28)	41(.943)	**21(.549)**
	CIFAR10	10000 × 3072	2000+(690)	2000+(697)	556(193)	2000+(669)	**94(31.7)**
	News20	19996 × 1355191	2000+(1.21e4)	2000+(9.16e3)	227(914)	104(391)	**71(287)**
	Rcv1	20242 × 47236	2000+(1.20e3)	1823(802)	196(79.1)	104(35.7)	**64(26.0)**
	Realsim	72309 × 20958	2000+(4.26e3)	2000+(4.33e3)	341(355)	152(125)	**107(88.2)**
Low rank least squares	Synthetic	1000 × 200	2000+(118)	2000+(116)	268(15.1)	26(1.55)	**18(1.04)**
	German	1000 × 24	2000+(4.72)	2000+(4.72)	642(1.52)	130(.334)	**52(.125)**
	Spectf	80 × 44	2000+(2.70)	2000+(2.74)	336(.455)	162(.236)	**105(.150)**
	MNIST	60000 × 784	200+(1.86e3)	200+(2.08e3)	200+(3.29e3)	200+(3.46e3)	**38(658)**
	CIFAR10	10000 × 3072	200+(7.24e3)	200+(1.33e4)	53(1.60e3)	8(208)	**6(156)**
QP and dual SVM	Synthetic	250 × 500	1224(11.5)	823(7.49)	626(5.93)	170(1.57)	**100(.914)**
	German	1000 × 24	2000+(58.8)	2000+(61.8)	1592(45.0)	1393(38.9)	**1238(34.9)**
	Spectf	80 × 44	2000+(.846)	2000+(.777)	169(.070)	175(.086)	**53(.026)**
Consensus logistic regression	Synthetic	1000 × 25	590(9.93)	391(6.97)	70(1.23)	35(.609)	**20(.355)**
	German	1000 × 24	2000+(34.3)	2000+(66.6)	151(2.60)	35(.691)	**26(.580)**
	Spectf	80 × 44	1005(20.1)	667(14.4)	117(1.98)	145(1.63)	**85(1.07)**
	MNIST	60000 × 784	200+(2.99e3)	200+(3.47e3)	200+(1.37e3)	49(536)	**28(333)**
	CIFAR10	10000 × 3072	200+(593)	200+(2.08e3)	200+(1.54e3)	131(165)	**19(33.7)**
Unwrapping SVM	Synthetic	1000 × 25	2000+(1.13)	1418(.844)	2000+(1.16)	355(.229)	**147(.094)**
	German	1000 × 24	753(1.88)	560(1.37)	2000+(4.98)	572(1.44)	**213(.545)**
	Spectf	80 × 44	567(.203)	367(.112)	567(.185)	207(.068)	**149(.052)**
	MNIST	60000 × 784	128(130)	118(111)	163(153)	200+(217)	**67(71.0)**
	CIFAR10	10000 × 3072	200+(512)	200+(532)	200+(516)	89(285)	**57(143)**
Image denoising	Barbara	512 × 512	262(35.0)	175(23.6)	74(10.0)	59(8.67)	**38(5.57)**
	Cameraman	256 × 256	311(8.96)	208(5.89)	82(2.29)	88(2.76)	**35(1.08)**
	Lena	512 × 512	347(46.3)	232(31.3)	94(12.5)	68(9.70)	**39(5.58)**
Robust PCA	FaceSet1	64 × 1024	2000+(41.1)	1507(30.3)	560(11.1)	561(11.9)	**267(5.65)**
	FaceSet2	64 × 1024	2000+(41.1)	2000+(41.4)	263(5.54)	388(9.00)	**188(4.02)**
	FaceSet3	64 × 1024	2000+(39.4)	1843(36.3)	375(7.44)	473(9.89)	**299(6.27)**

[1] #constrains × #unknowns for canonical QP; width × height for image restoration.

where the nuclear norm $\|\cdot\|_*$ is used to obtain a low rank matrix Z, and $\|\cdot\|_1$ is used to obtain a sparse error E.

7. Experiments

The proposed AADMM is implemented as shown in Algorithm 1. We also implemented vanilla ADMM, (non-adaptive) relaxed ADMM, ADMM with residual balancing (RB), and adaptive ADMM (AADMM) for comparison.

The relaxation parameter for the non-adaptive relaxed ADMM is fixed at $\gamma_k = 1.5$ as suggested in [6]. The parameters of RB and AADMM are selected as in [24, 2, 51]. The initial penalty $\tau_0 = 1/10$ and initial relaxation $\gamma_0 = 1$ are used for all problems except the canonical QP problem, where initial parameters are set to the geometric mean of the maximum and minimum eigenvalues of matrix Q, as proposed for quadratic problems in [40].

For each problem, the same randomly generated initial variables v_0, λ_0 are used for ADMM and its variant methods. As suggested by [24, 51], the adaptivity of RB and AADMM is stopped after 1000 iterations to guarantee convergence.

7.1. Convergence results

Table 1 reports the convergence speed of ADMM and its variants for the applications described in Section 6. More experimental results including the table of more test cases, the convergence curves, and visual results of image restoration and robust PCA for face decomposition are provided in the supplementary material. Relaxed ADMM often outperforms vanilla ADMM, but does not compete with adaptive methods like RB, AADMM and ARADMM. The proposed ARADMM performs best in all the test cases.

7.2. Sensitivity to initialization

We study the sensitivity of the different ADMM variants to the initial penalty (τ_0) and initial relaxation parameter (γ_0). Fig. 1 presents iteration counts for a wide range of values of τ_0, γ_0, for elastic net regression with synthetic datasets. In the left and center plots we fix one of τ_0, γ_0 and vary the other. The number of iterations needed to convergence is plotted as the algorithm parameters vary. In the right plot, we use a grid search to find the *optimal* τ_0 for different

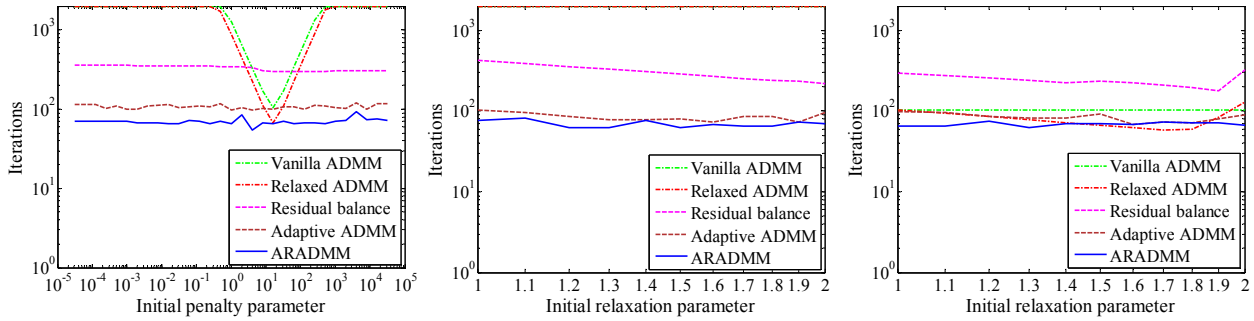

Figure 1. Sensitivity of convergence speed for the synthetic problem of EN regularized linear regression. (left) sensitivity to the initial penalty τ_0; (middle) sensitivity to relaxation γ_0; (right) sensitivity to relaxation γ_0 when optimal τ_0 is selected by grid search.

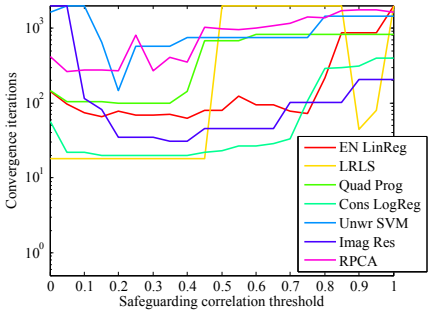

Figure 2. Sensitivity of convergence speed to safeguarding threshold ϵ^{cor} for proposed ARADMM. Synthetic problems ('cameraman' for TVIR, and 'FaceSet1' for RPCA) of various applications are studied. Best viewed in color.

values of γ_0. Fig. 1 (left) shows that adaptive methods are relatively stable with respect to the initial penalty τ_0, while ARADMM outperforms RB and AADMM in all choices of initial τ_0. Fig. 1 (middle) suggests that the relaxation γ_0 is generally less important than τ_0. When a bad value of τ is chosen, it is unlikely that a good choice of γ can compensate. The proposed ARADMM that jointly adjusts τ, γ is generally better than simply adding the relaxation to the existing adaptive methods RB and AADMM.

Fig. 1 (right) shows the sensitivity to γ when using a grid search to choose the optimal τ_0. This optimal τ_0 significantly improves the performance of vanilla ADMM and relaxed ADMM (which use the same τ_0 for all iterations). Even when using the optimal stepsize for the non-adaptive methods, ARADMM is superior to or competitive with the non-adaptive methods. Note that this experiment is meant to show a best-case scenario for the non-adaptive methods; in practice the user generally has no knowledge of the optimal value of τ. Adaptive methods achieve optimal or near-optimal performance without an expensive grid search.

7.3. Sensitivity to safeguarding

Finally, Fig. 2 presents iteration counts when applying ARADMM with various safeguarding correlation thresholds ϵ^{cor}. When $\epsilon^{\text{cor}} = 0$, the calculated adaptive parameters based on curvature estimations are always accepted, and when $\epsilon^{\text{cor}} = 1$ the parameters are never changed. The proposed AADMM method is insensitive to ϵ^{cor} and performs well for a wide range of $\epsilon^{\text{cor}} \in [0.1,\ 0.4]$ for various applications, except for unwrapping SVM and RPCA. Though tuning such "hyper-parameters" may improve the performance of ARADMM for some applications, the fixed $\epsilon^{\text{cor}} = 0.2$ performs well in all our experiments (seven applications and over fifty test cases, a full list is in the supplementary material). The proposed ARADMM is fully automated and performs well without parameter tuning.

8. Conclusion

We have proposed an adaptive method for jointly tuning the penalty and relaxation parameters of relaxed ADMM without user oversight. We have analyzed adaptive relaxed ADMM schemes, and provided conditions for which convergence is guaranteed. Experiments on a wide range of machine learning, computer vision, and image processing benchmarks have demonstrated that the proposed adaptive method (often significantly) outperforms other ADMM variants without user oversight or parameter tuning. The new adaptive method improves the applicability of relaxed ADMM by facilitating fully automated solvers that exhibit fast convergence and are usable by non-expert users.

Acknowledgments

TG and ZX were supported by the US Office of Naval Research under grant N00014-17-1-2078 and by the US National Science Foundation (NSF) under grant CCF-1535902. MF was partially supported by the Fundação para a Ciência e Tecnologia, grant UID/EEA/5008/2013. XY was supported by the General Research Fund from Hong Kong Research Grants Council under grant HKBU-12313516. CS was supported in part by Xilinx Inc., and by the US NSF under grants ECCS-1408006, CCF-1535897, and CAREER CCF-1652065.

References

[1] J. Barzilai and J. Borwein. Two-point step size gradient methods. *IMA J. Num. Analysis*, 8:141–148, 1988. 3

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. and Trends in Mach. Learning*, 3:1–122, 2011. 1, 2, 6, 7

[3] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011. 1, 6

[4] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 1

[5] D. Davis and W. Yin. Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions. *arXiv preprint arXiv:1407.5210*, 2014. 2, 3

[6] J. Eckstein and D. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, 1992. 1, 2, 3, 7

[7] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004. 2, 6

[8] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2790–2797. IEEE, 2009. 1, 2

[9] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008. 2

[10] E. X. Fang, B. He, H. Liu, and X. Yuan. Generalized alternating direction method of multipliers: new theoretical insights and applications. *Mathematical Programming Computation*, 7(2):149–187, 2015. 1, 2

[11] R. Fletcher. On the Barzilai-Borwein method. In *Optimization and control with applications*, pages 235–256. Springer, 2005. 3

[12] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976. 1

[13] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE transactions on pattern analysis and machine intelligence*, 23(6):643–660, 2001. 6

[14] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson. Optimal parameter selection for the alternating direction method of multipliers: quadratic problems. *IEEE Trans. Autom. Control*, 60:644–658, 2015. 1, 2

[15] P. Giselsson and S. Boyd. Linear convergence and metric selection in Douglas-Rachford splitting and ADMM. 2016. 2, 3

[16] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problémes de Dirichlet non linéaires. *ESAIM: Modélisation Mathématique et Analyse Numérique*, 9:41–76, 1975. 1

[17] D. Goldfarb, S. Ma, and K. Scheinberg. Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming*, 141(1-2):349–382, 2013. 2

[18] D. Goldfarb and Z. Qin. Robust low-rank tensor recovery: Models and algorithms. *SIAM Journal on Matrix Analysis and Applications*, 35(1):225–253, 2014. 2

[19] T. Goldstein, P. Hand, C. Lee, V. Voroninski, and S. Soatto. Shapefit and shapekick for robust, scalable structure from motion. In *European Conference on Computer Vision*, pages 289–304. Springer, 2016. 2

[20] T. Goldstein, M. Li, and X. Yuan. Adaptive primal-dual splitting methods for statistical learning and image processing. In *Advances in Neural Information Processing Systems*, pages 2080–2088, 2015. 1

[21] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014. 1, 2, 6

[22] T. Goldstein, G. Taylor, K. Barabin, and K. Sayre. Unwrapping ADMM: efficient distributed computing via transpose reduction. In *AISTATS*, 2016. 6

[23] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick. Large-scale image classification with trace-norm regularization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3386–3393. IEEE, 2012. 1, 2

[24] B. He, H. Yang, and S. Wang. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Jour. Optim. Theory and Appl.*, 106(2):337–356, 2000. 1, 2, 7

[25] B. He and X. Yuan. On the o(1/n) convergence rate of the Douglas-Rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2):700–709, 2012. 1, 2

[26] B. He and X. Yuan. On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers. *Numerische Mathematik*, 130:567–577, 2015. 1, 2

[27] M. Kadkhodaie, K. Christakopoulou, M. Sanjabi, and A. Banerjee. Accelerated alternating direction method of multipliers. In *ACM SIGKDD*, pages 497–506, 2015. 2

[28] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 6

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[30] S.-I. Lee, H. Lee, P. Abbeel, and A. Ng. Efficient L1 regularized logistic regression. In *AAAI*, volume 21, page 401, 2006. 6

[31] W. Li, Z. Xu, D. Xu, D. Dai, and L. V. Gool. Domain generalization and adaptation using low rank exemplar svms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. 1, 2

[32] Z. Lin, R. Liu, and Z. Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In *NIPS*, pages 612–620, 2011. 1

[33] J. Liu, J. Chen, and J. Ye. Large-scale sparse logistic regression. In *ACM SIGKDD*, pages 547–556, 2009. 2, 6

[34] R. Liu, Z. Lin, and Z. Su. Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning. In *ACML*, pages 116–132, 2013. 1

[35] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan. Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2

[36] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. *Foundations and Trends® in Computer Graphics and Vision*, 8(2-3):85–283, 2014. 1, 2

[37] N. Naikal, A. Y. Yang, and S. S. Sastry. Informative feature selection for object recognition via sparse PCA. In *2011 International Conference on Computer Vision*, pages 818–825. IEEE, 2011. 6

[38] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan. A general analysis of the convergence of ADMM. In *ICML*, 2015. 2

[39] O. Ozyesil and A. Singer. Robust camera location estimation by convex programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2674–2683, 2015. 6

[40] A. Raghunathan and S. Di Cairano. Alternating direction method of multipliers for strictly convex quadratic programs: Optimal parameter selection. In *American Control Conf.*, pages 4324–4329, 2014. 1, 2, 7

[41] R. Rockafellar. *Convex Analysis*. Princeton University Press, 1970. 3

[42] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992. 2, 6

[43] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *ECML*, pages 286–297. Springer, 2007. 2, 6

[44] C. Song, S. Yoon, and V. Pavlovic. Fast ADMM algorithm for distributed optimization with adaptive penalty. *arXiv preprint arXiv:1506.08928*, 2015. 1

[45] G. Taylor, R. Burmeister, Z. Xu, B. Singh, A. Patel, and T. Goldstein. Training neural networks without gradients: A scalable ADMM approach. *ICML*, 2016. 2

[46] W. Tian and X. Yuan. Faster alternating direction method of multipliers with a worst-case o $(1/n^2)$ convergence rate. 2016. 2

[47] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, pages 2080–2088, 2009. 1, 2, 6

[48] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31:210–227, 2009. 1, 2

[49] S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Signal Processing*, 57:2479–2493, 2009. 3

[50] Z. Xu, S. De, M. A. T. Figueiredo, C. Studer, and T. Goldstein. An empirical study of ADMM for nonconvex problems. In *NIPS workshop on nonconvex optimization*, 2016. 2

[51] Z. Xu, M. A. Figueiredo, and T. Goldstein. Adaptive ADMM with spectral penalty parameter selection. *AISTATS*, 2017. 1, 2, 3, 4, 7

[52] Z. Xu, F. Huang, L. Raschid, and T. Goldstein. Non-negative factorization of the occurrence tensor from financial contracts. In *NIPS workshop on tensor methods*, 2016. 2

[53] Z. Xu, X. Li, K. Yang, and T. Goldstein. Exploiting low-rank structure for discriminative sub-categorization. In *BMVC, Swansea, UK, September 7-10, 2015*, 2015. 1, 2, 6

[54] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009. 1, 2

[55] J. Yang and X. Yuan. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation*, 82(281):301–329, 2013. 6

[56] Z. Zhang, Y. Chen, and V. Saligrama. Efficient training of very deep neural networks for supervised hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1487–1495, 2016. 2

[57] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005. 2, 6