Adaptive Consensus ADMM for Distributed Optimization

Zheng Xu [1] **Gavin Taylor** [2] **Hao Li** [1] **Mário A. T. Figueiredo** [3] **Xiaoming Yuan** [4] **Tom Goldstein** [1]

Abstract

The alternating direction method of multipliers (ADMM) is commonly used for distributed model fitting problems, but its performance and reliability depend strongly on user-defined penalty parameters. We study distributed ADMM methods that boost performance by using different fine-tuned algorithm parameters on each worker node. We present a $O(1/k)$ convergence rate for adaptive ADMM methods with node-specific parameters, and propose *adaptive consensus ADMM (ACADMM)*, which automatically tunes parameters without user oversight.

1. Introduction

The alternating direction method of multipliers (ADMM) is a popular tool for solving problems of the form,

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

where $f:\mathbb{R}^n\to\mathbb{R}$ and $g:\mathbb{R}^m\to\mathbb{R}$ are 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 (Glowinski & Marroco, 1975) and (Gabay & Mercier, 1976), and has found applications in many optimization problems in machine learning, distributed computing and many other areas (Boyd et al., 2011).

Consensus ADMM (Boyd et al., 2011) solves minimization problems involving a composite objective $f(v)=\sum_i f_i(v)$, where worker i stores the data needed to compute f_i, and so is well suited for distributed model fitting problems (Boyd et al., 2011; Zhang & Kwok, 2014; Song et al., 2016; Chang et al., 2016; Goldstein et al., 2016; Taylor et al., 2016). To distribute this problem, consensus methods assign a separate copy of the unknowns, u_i, to each worker, and then apply ADMM to solve

$$\min_{u_i\in\mathbb{R}^d, v\in\mathbb{R}^d} \sum_{i=1}^N f_i(u_i)+g(v), \quad \text{subject to } u_i=v, \quad (2)$$

where v is the "central" copy of the unknowns, and $g(v)$ is a regularizer. The consensus problem (2) coincides with (1) by defining $u=(u_1;\ldots;u_N)\in\mathbb{R}^{dN}$, $A=I_{dN}\in\mathbb{R}^{dN\times dN}$, and $B=-(I_d;\ldots;I_d)\in\mathbb{R}^{dN\times d}$, where I_d represents the $d\times d$ identity matrix.

ADMM methods rely on a penalty parameter (stepsize) that is chosen by the user. In theory, ADMM converges for any constant penalty parameter (Eckstein & Bertsekas, 1992; He & Yuan, 2012; Ouyang et al., 2013). In practice, however, the efficiency of ADMM is highly sensitive to this parameter choice (Nishihara et al., 2015; Ghadimi et al., 2015), and can be improved via adaptive penalty selection methods (He et al., 2000; Song et al., 2016; Xu et al., 2017a).

One such approach, residual balancing (RB) (He et al., 2000), adapts the penalty parameter so that the residuals (derivatives of the Lagrangian with respect to primal and dual variables) have similar magnitudes. When the same penalty parameter is used across nodes, RB is known to converge, although without a known rate guarantee. A more recent approach, AADMM (Xu et al., 2017a), achieves impressive practical convergence speed on many applications, including consensus problems, with adaptive penalty parameters by estimating the local curvature of the dual functions. However, the dimension of the unknown variables in consensus problems grows with the number of distributed nodes, causing the curvature estimation to be inaccurate and unstable. AADMM uses the same convergence analysis as RB. Consensus residual balancing (CRB) (Song et al., 2016) extends residual balancing to consensus-based ADMM for distributed optimization by balancing the local primal and dual residuals on each node. However, convergence guarantees for this method are fairly weak, and adaptive penalties need to be reset after several iterations to guarantee convergence.

We study the use of adaptive ADMM in the distributed setting, where different workers use different local algorithm parameters to accelerate convergence. We begin by studying the theory and provide convergence guarantees when

[1]University of Maryland, College Park; [2]United States Naval Academy, Annapolis; [3]Instituto de Telecomunicações, IST, ULisboa, Portugal; [4]Hong Kong Baptist University, Hong Kong. Correspondence to: Zheng Xu <xuzhustc@gmail.com>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

node-specific penalty parameters are used. We demonstrate a $O(1/k)$ convergence rate under mild conditions that is applicable for many forms of adaptive ADMM including all the above methods. Our theory is more general than the convergence guarantee in (He et al., 2000; Xu et al., 2017a) that only shows convergence when the scalar penalty parameter is adapted. Next, we propose an adaptive consensus ADMM (ACADMM) method to automate local algorithm parameters selection. Instead of estimating one global penalty parameter for all workers, different local penalty parameters are estimated using the local curvature of subproblems on each node.

2. Related work

ADMM is known to have a $O(1/k)$ convergence rate under mild conditions for convex problems (He & Yuan, 2012; 2015), while a $O(1/k^2)$ rate is possible when at least one of the functions is strongly convex or smooth (Goldfarb et al., 2013; Goldstein et al., 2014; Kadkhodaie et al., 2015; Tian & Yuan, 2016). Linear convergence can be achieved with strong convexity assumptions (Davis & Yin, 2014; Nishihara et al., 2015; Giselsson & Boyd, 2016). All of these results assume constant parameters; to the best of our knowledge, no convergence rate has been proven for ADMM with an adaptive penalty: (He et al., 2000; Xu et al., 2017b) proves convergence without providing a rate, and (Lin et al., 2011; Banert et al., 2016; Goldstein et al., 2015) prove convergence for some particular variants of ADMM ("linearized" or "preconditioned").

To improve practical convergence of ADMM, fixed optimal parameters are discussed in (Raghunathan & Di Cairano, 2014; Ghadimi et al., 2015; Nishihara et al., 2015; França & Bento, 2016). These methods make strong assumptions about the objective and require information about the spectrum of A and/or B. Additionally, adaptive methods have been proposed; the most closely related work to our own is (Song et al., 2016), which extends the results of (He et al., 2000) to consensus problems, where communication is controlled by predefined network structure and the regularizer $g(v)$ is absent. In contrast to these methods, the proposed ACADMM extends the spectral penalty in (Xu et al., 2017a) to consensus problems and provides convergence theory that can be applied to a broad range of adaptive ADMM variants.

3. Consensus ADMM

In the following, we use the subscript i to denote iterates computed on the ith node, superscript k is the iteration number, λ_i^k is the dual vector of Lagrange multipliers, and $\{\tau_i^k\}$ are iteration/worker-specific penalty parameters (contrasted with the single constant penalty parameter τ of "vanilla" ADMM). Consensus methods apply ADMM to (2), resulting in the steps

$$u_i^{k+1} = \arg\min_{u_i} f_i(u_i) + \frac{\tau_i^k}{2}\|v^k - u_i + \frac{\lambda_i^k}{\tau_i^k}\|^2 \tag{3}$$

$$v^{k+1} = \arg\min_{v} g(v) + \sum_{i=1}^{N} \frac{\tau_i^k}{2}\|v - u_i^{k+1} + \frac{\lambda_i^k}{\tau_i^k}\|^2 \tag{4}$$

$$\lambda_i^{k+1} = \lambda_i^k + \tau_i^k(v^{k+1} - u_i^{k+1}). \tag{5}$$

The primal and dual residuals, r^k and d^k, are used to monitor convergence.

$$r^k = \begin{pmatrix} r_1^k \\ \vdots \\ r_N^k \end{pmatrix},\ d^k = \begin{pmatrix} d_1^k \\ \vdots \\ d_N^k \end{pmatrix},\ \begin{cases} r_i^k = v^k - u_i^k \\ d_i^k = \tau_i^k(v^{k-1} - v^k). \end{cases} \tag{6}$$

The primal residual r^k approaches zero when the iterates accurately satisfy the linear constraints in (2), and the dual residual d^k approaches zero as the iterates near a minimizer of the objective. Iteration can be terminated when

$$\begin{aligned} &\|r^k\|^2 \le \epsilon^{tol} \max\{\textstyle\sum_{i=1}^N \|u_i^k\|^2, N\|v^k\|^2\} \\ &\text{and } \|d^k\|^2 \le \epsilon^{tol} \textstyle\sum_{i=1}^N \|\lambda_i^k\|^2, \end{aligned} \tag{7}$$

where ϵ^{tol} is the stopping tolerance. The residuals in (6) and stopping criterion in (7) are adopted from the general problem (Boyd et al., 2011) to the consensus problem. The observation that residuals r^k, d^k can be decomposed into "local residuals" r_i^k, d_i^k has been exploited to generalize the residual balancing method (He et al., 2000) for distributed consensus problems (Song et al., 2016).

4. Convergence analysis

We now study the convergence of ADMM with node-specific adaptive penalty parameters. We provide conditions on penalty parameters that guarantee convergence, and also a convergence rate. The issue of how to automatically tune penalty parameters effectively will be discussed in Section 5.

4.1. Diagonal penalty parameters for ADMM

Let $T^k = \text{diag}(\tau_1^k I_d, \ldots, \tau_N^k I_d)$ be a diagonal matrix containing non-negative penalty parameters on iteration k. Define the norm $\|u\|_T^2 = u^T T u$. Using the notation defined above with $u = (u_1; \ldots; u_N) \in \mathbb{R}^{dN}$, we can rewrite the consensus ADMM steps (3)–(5) as

$$\begin{aligned} u^{k+1} = \arg\min_u\ & f(u) + \langle -Au,\ \lambda^k \rangle \\ & + 1/2\|b - Au - Bv^k\|_{T^k}^2 \end{aligned} \tag{8}$$

$$v^{k+1} = \arg\min_v g(v) + \langle -Bv,\ \lambda^k \rangle + {}^1\!/_2 \|b - Au^{k+1} - Bv\|^2_{T^k} \tag{9}$$

$$\lambda^{k+1} = \lambda^k + T^k(b - Au^{k+1} - Bv^{k+1}). \tag{10}$$

When using a diagonal penalty matrix, the generalized residuals become

$$\begin{cases} r^k = b - Au^k - Bu^k \\ d^k = A^T T^k B(v^k - v^{k-1}). \end{cases} \tag{11}$$

The sequel contains a convergence proof for generalized ADMM with adaptive penalty matrix T^k. Our proof is inspired by the variational inequality (VI) approach in (He et al., 2000; He & Yuan, 2012; 2015).

4.2. Preliminaries

Notation. We use the following notation to simplify the discussions. Define the combined variables $y = (u; v) \in \mathbb{R}^{n+m}$ and $z = (u; v; \lambda) \in \mathbb{R}^{n+m+p}$, and denote iterates as $y^k = (u^k; v^k)$ and $z^k = (u^k; v^k; \lambda^k)$. Let y^* and z^* denote optimal primal/dual solutions. Further define $\Delta z_k^+ = (\Delta u_k^+; \Delta v_k^+; \Delta\lambda_k^+) := z^{k+1} - z^k$ and $\Delta z_k^* = (\Delta u_k^*; \Delta v_k^*; \Delta\lambda_k^*) := z^* - z^k$. Set

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

$$H^k = \begin{pmatrix} 0 & 0 & 0 \\ 0 & B^T T^k B & 0 \\ 0 & 0 & (T^k)^{-1} \end{pmatrix},\ M^k = \begin{pmatrix} I_n & 0 & 0 \\ 0 & I_m & 0 \\ 0 & -T^k B & I_p \end{pmatrix}.$$

Note that $F(z)$ is a monotone operator satisfying $\forall z, z', (z - z')^T(F(z) - F(z')) \geq 0$. We introduce intermediate variable $\tilde{z}^{k+1} = (u^{k+1}; v^{k+1}; \hat{\lambda}^{k+1})$, where $\hat{\lambda}^{k+1} = \lambda^k + T^k(b - Au^{k+1} - Bv^k)$. We thus have

$$\Delta z_k^+ = M^k(\tilde{z}^{k+1} - z^k). \tag{12}$$

Variational inequality formulation. The optimal solution z^* of problem (1) satisfies the variational inequality (VI),

$$\forall z,\ \phi(y) - \phi(y^*) + (z - z^*)^T F(z^*) \geq 0. \tag{13}$$

From the optimality conditions for the sub-steps (8, 9), we see that y^{k+1} satisfies the variational inequalities

$$\forall u, f(u) - f(u^{k+1}) + (u - u^{k+1})^T (A^T T^k(Au^{k+1} + Bv^k - b) - A^T\lambda^k) \geq 0 \tag{14}$$

$$\forall v, g(v) - g(v^{k+1}) + (v - v^{k+1})^T (B^T T^k(Au^{k+1} + Bv^{k+1} - b) - B^T\lambda^k) \geq 0, \tag{15}$$

which can be combined as

$$\phi(y) - \phi(y^{k+1}) + (z - \tilde{z}^{k+1})^T \left(F(\tilde{z}^{k+1}) + H^k \Delta z_k^+\right) \geq 0. \tag{16}$$

Lemmas. We present several lemmas to facilitate the proof of our main convergence theory, which extend previous results regarding ADMM (He & Yuan, 2012; 2015) to ADMM with a diagonal penalty matrix. Lemma 1 shows the difference between iterates decreases as the iterates approach the true solution, while Lemma 2 implies a contraction in the VI sense. Full proofs are provided in supplementary material; Eq. (17) and Eq. (18) are supported using equations (13, 15, 16) and standard techniques, while Eq. (19) is proven from Eq. (18). Lemma 2 is supported by the relationship in Eq. (12).

Lemma 1. *The optimal solution $z^* = (u^*; v^*; \lambda^*)$ and sequence $z^k = (u^k; v^k; \lambda^k)$ of generalized ADMM satisfy*

$$(B\Delta v_k^+)^T \Delta\lambda_k^+ \geq 0, \tag{17}$$

$$\Delta z_{k+1}^* H^k \Delta z_k^+ \geq 0, \tag{18}$$

$$\|\Delta z_k^+\|^2_{H^k} \leq \|\Delta z_k^*\|^2_{H^k} - \|\Delta z_{k+1}^*\|^2_{H^k}. \tag{19}$$

Lemma 2. *The sequence $\tilde{z}^k = (u^k; v^k; \hat{\lambda}^k)$ and $z^k = (u^k; v^k; \lambda^k)^T$ from generalized ADMM satisfy, $\forall z$,*

$$(\tilde{z}^{k+1} - z)^T H^k \Delta z_k^+ \geq \frac{1}{2}(\|z^{k+1} - z\|^2_{H^k} - \|z^k - z\|^2_{H^k}). \tag{20}$$

4.3. Convergence criteria

We provide a convergence analysis of ADMM with an adaptive diagonal penalty matrix by showing (i) the norm of the residuals converges to zero; (ii) the method attains a worst-case ergodic $O(1/k)$ convergence rate in the VI sense. The key idea of the proof is to bound the adaptivity of T^k so that ADMM is stable enough to converge, which is presented as the following assumption.

Assumption 1. *The adaptivity of the diagonal penalty matrix $T^k = diag(\tau_i^k, \ldots, \tau_p^k)$ is bounded by*

$$\sum_{k=1}^{\infty} (\eta^k)^2 < \infty,\ \textit{where}\ (\eta^k)^2 = \max_{i \in \{1,\ldots,p\}} \{(\eta_i^k)^2\},\ (\eta_i^k)^2 = \max\{\tau_i^k/\tau_i^{k-1} - 1, \tau_i^{k-1}/\tau_i^k - 1\}. \tag{21}$$

We can apply Assumption 1 to verify that

$$\frac{1}{1 + (\eta^k)^2} \leq \frac{\tau_i^k}{\tau_i^{k-1}} \leq 1 + (\eta^k)^2. \tag{22}$$

which is needed to prove Lemma 3.

Lemma 3. *Suppose Assumption 1 holds. Then $z = (u;\ v;\ \lambda)$ and $z' = (u';\ v';\ \lambda')$ satisfy, $\forall z, z'$*

$$\|z - z'\|^2_{H^k} \leq (1 + (\eta^k)^2)\|z - z'\|^2_{H^{k-1}}. \tag{23}$$

Now we are ready to prove the convergence of generalized ADMM with adaptive penalty under Assumption 1. We prove the following quantity, which is a norm of the residuals, converges to zero.

$$\begin{aligned}\|\Delta z_k^+\|_{H^k}^2 =&\|B\Delta v_k^+\|_{T^k}^2+\|\Delta\lambda_k^+\|_{(T^k)^{-1}}^2\\ =&\|(A^TT^k)^\dagger d^k\|_{T^k}^2+\|r^k\|_{T^k}^2,\end{aligned}\tag{24}$$

where $A^\dagger$ denotes generalized inverse of a matrix A. Note that $\|\Delta z_k^+\|_{H^k}^2$ converges to zero only if $\|r^k\|$ and $\|d^k\|$ converge to zero, provided A and T^k are bounded.

Theorem 1. *Suppose Assumption 1 holds. Then the iterates $z^k=(u^k;v^k;\lambda^k)$ of generalized ADMM satisfy*

$$\lim_{k\to\infty}\|\Delta z_k^+\|_{H^k}^2=0.\tag{25}$$

Proof. Let $z=z^k, z'=z^*$ in Lemma 3 to achieve

$$\|\Delta z_k^*\|_{H^k}^2\le(1+(\eta^k)^2)\|\Delta z_k^*\|_{H^{k-1}}^2.\tag{26}$$

Combine (26) with Lemma 1 (19) to get

$$\|\Delta z_k^+\|_{H^k}^2\le(1+(\eta^k)^2)\|\Delta z_k^*\|_{H^{k-1}}^2-\|\Delta z_{k+1}^*\|_{H^k}^2.\tag{27}$$

Accumulate (27) for $k=1$ to l,

$$\begin{aligned}&\sum_{k=1}^l\prod_{t=k+1}^l(1+(\eta^t)^2)\|\Delta z_k^+\|_{H^k}^2\le\\ &\prod_{t=1}^l(1+(\eta^t)^2)\|\Delta z_1^*\|_{H^0}^2-\|\Delta z_{l+1}^*\|_{H^l}^2.\end{aligned}\tag{28}$$

Then we have

$$\sum_{k=1}^l\|\Delta z_k^+\|_{H^k}^2\le\prod_{t=1}^l(1+(\eta^t)^2)\|\Delta z_1^*\|_{H^0}^2.\tag{29}$$

When $l\to\infty$, Assumption 1 suggests $\prod_{t=1}^\infty(1+(\eta^t)^2)<\infty$, which means $\sum_{k=1}^\infty\|\Delta z_k^+\|_{H^k}^2<\infty$. Hence $\lim_{k\to\infty}\|\Delta z_k^+\|_{H^k}^2=0$. □

We further exploit Assumption 1 and Lemma 3 to prove Lemma 4, and combine VI (16), Lemma 2, and Lemma 4 to prove the $O(1/k)$ convergence rate in Theorem 2.

Lemma 4. *Suppose Assumption 1 holds. Then $z=(u;v;\lambda)\in\mathbb{R}^{m+n+p}$ and the iterates $z^k=(u^k;v^k;\lambda^k)$ of generalized ADMM satisfy, $\forall z$*

$$\begin{aligned}&\sum_{k=1}^l(\|z-z^k\|_{H^k}^2-\|z-z^k\|_{H^{k-1}}^2)\le\\ &C_\eta^\Sigma C_\eta^\Pi(\|z-z^*\|_{H^0}^2+\|\Delta z_1^*\|_{H^0}^2)<\infty,\end{aligned}\tag{30}$$

where $C_\eta^\Sigma=\sum_{k=1}^\infty(\eta^k)^2$, $C_\eta^\Pi=\prod_{t=1}^\infty(1+(\eta^t)^2)$.

Theorem 2. *Suppose Assumption 1 holds. Consider the sequence $\tilde z^k=(u^k;v^k;\hat\lambda^k)$ of generalized ADMM and define $\bar z^l=\frac{1}{l}\sum_{k=1}^l\tilde z^k$. Then sequence $\bar z^l$ satisfies the convergence bound*

$$\begin{aligned}\phi(y)-\phi(\bar y^l)+(z-\bar z^l)^TF(\bar z^l)\ge-\frac{1}{2\,l}(&\|z-z^0\|_{H^0}^2\\ &+C_\eta^\Sigma C_\eta^\Pi\|z-z^*\|_{H^0}^2+C_\eta^\Sigma C_\eta^\Pi\|\Delta z_1^*\|_{H^0}^2).\end{aligned}\tag{31}$$

Proof. We can verify with simple algebra that

$$(z-z')^TF(z)=(z-z')^TF(z').\tag{32}$$

Apply (32) with $z'=\tilde z^{k+1}$, and combine VI (16) and Lemma 2 to get

$$\begin{aligned}&\phi(y)-\phi(y^{k+1})+(z-\tilde z^{k+1})^TF(z)&(33)\\ =&\phi(y)-\phi(y^{k+1})+(z-\tilde z^{k+1})^TF(\tilde z^{k+1})&(34)\\ \ge&(\tilde z^{k+1}-z)^TH^k\Delta z_k^+&(35)\\ \ge&\frac{1}{2}(\|z^{k+1}-z\|_{H^k}^2-\|z^k-z\|_{H^k}^2).&(36)\end{aligned}$$

Summing for $k=0$ to $l-1$ gives us

$$\begin{aligned}&\sum_{k=1}^l\phi(y)-\phi(y^k)+(z-\tilde z^k)^TF(z)\\ \ge&\frac{1}{2}\sum_{k=1}^l(\|z-z^k\|_{H^{k-1}}^2-\|z-z^{k-1}\|_{H^{k-1}}^2).\end{aligned}\tag{37}$$

Since $\phi(y)$ is convex, the left hand side of (37) satisfies,

$$\begin{aligned}LHS=&\,l\,\phi(y)-\sum_{k=1}^l\phi(y^k)+(l\,z-\sum_{k=1}^l\tilde z^k)^TF(z)\\ \le&\,l\,\phi(y)-l\,\phi(\bar y^l)+(l\,z-l\,\bar z^l)^TF(z).\end{aligned}\tag{38}$$

Applying Lemma 4, we see the right hand side satisfies,

$$\begin{aligned}RHS=&\frac{1}{2}\sum_{k=1}^l(\|z-z^k\|_{H^k}^2-\|z-z^{k-1}\|_{H^{k-1}}^2)+\\ &\frac{1}{2}\sum_{k=1}^l(\|z-z^k\|_{H^{k-1}}^2-\|z-z^k\|_{H^k}^2)\end{aligned}\tag{39}$$

$$\begin{aligned}\ge&\frac{1}{2}(\|z-z^l\|_{H^l}^2-\|z-z^0\|_{H^0}^2)+\\ &-\frac{1}{2}C_\eta^\Sigma C_\eta^\Pi(\|z-z^*\|_{H^0}^2+\|\Delta z_1^*\|_{H^0}^2)\end{aligned}\tag{40}$$

$$\begin{aligned}\ge-&\frac{1}{2}(\|z-z^0\|_{H^0}^2+C_\eta^\Sigma C_\eta^\Pi\|z-z^*\|_{H^0}^2+\\ &C_\eta^\Sigma C_\eta^\Pi\|\Delta z_1^*\|_{H^0}^2).\end{aligned}\tag{41}$$

Combining inequalities (37), (38) and (41), and letting $z'=\bar z^k$ in (32) yields the $O(1/k)$ convergence rate in (31) □

5. Adaptive Consensus ADMM (ACADMM)

To address the issue of how to automatically tune parameters on each node for optimal performance, we propose *adaptive consensus ADMM* (ACADMM), which sets worker-specific penalty parameters by exploiting curvature information. We derive our method from the dual interpretation of ADMM – *Douglas-Rachford splitting* (DRS) – using a diagonal penalty matrix. We then derive the spectral stepsizes for consensus problems by assuming the curvatures of the objectives are diagonal matrices with diverse parameters on different nodes. At last, we discuss the practical computation of the spectral stepsizes from consensus ADMM iterates and apply our theory in Section 4 to guarantee convergence.

5.1. Dual interpretation of generalized ADMM

The dual form of problem (1) can be written

$$\min_{\lambda\in\mathbb{R}^p} \underbrace{f^*(A^T\lambda) - \langle \lambda, b\rangle}_{\hat{f}(\lambda)} + \underbrace{g^*(B^T\lambda)}_{\hat{g}(\lambda)}, \tag{42}$$

where λ denotes the dual variable, while f^*, g^* denote the Fenchel conjugate of f, g (Rockafellar, 1970). It is known that ADMM steps for the primal problem (1) are equivalent to performing *Douglas-Rachford splitting* (DRS) on the dual problem (42) (Eckstein & Bertsekas, 1992; Xu et al., 2017a). In particular, the generalized ADMM iterates satisfy the DRS update formulas

$$0 \in (T^k)^{-1}(\hat{\lambda}^{k+1} - \lambda^k) + \partial\hat{f}(\hat{\lambda}^{k+1}) + \partial\hat{g}(\lambda^k) \tag{43}$$

$$0 \in (T^k)^{-1}(\lambda^{k+1} - \lambda^k) + \partial\hat{f}(\hat{\lambda}^{k+1}) + \partial\hat{g}(\lambda^{k+1}), \tag{44}$$

where $\hat{\lambda}$ denotes the intermediate variable defined in Section 4.2. We prove the equivalence of generalized ADMM and DRS in the supplementary material.

5.2. Generalized spectral stepsize rule

Xu et al. (2017a) first derived spectral penalty parameters for ADMM using the DRS. Proposition 1 in (Xu et al., 2017a) proved that the minimum residual of DRS can be obtained by setting the scalar penalty to $\tau^k = 1/\sqrt{\alpha\,\beta}$, where we assume the subgradients are locally linear as

$$\partial\hat{f}(\hat{\lambda}) = \alpha\,\hat{\lambda} + \Psi \quad \text{and} \quad \partial\hat{g}(\lambda) = \beta\,\lambda + \Phi, \tag{45}$$

$\alpha, \beta \in \mathbb{R}$ represent scalar curvatures, and $\Psi, \Phi \subset \mathbb{R}^p$.

We now present generalized spectral stepsize rules that can accomodate consensus problems.

Proposition 1 (Generalized spectral DRS)**.** *Suppose the generalized DRS steps (43, 44) are used, and assume the subgradients are locally linear,*

$$\partial\hat{f}(\hat{\lambda}) = M_\alpha\,\hat{\lambda} + \Psi \quad \textit{and} \quad \partial\hat{g}(\lambda) = M_\beta\,\lambda + \Phi. \tag{46}$$

for matrices $M_\alpha = diag(\alpha_1 I_d, \ldots, \alpha_N I_d)$ and $M_\beta = diag(\beta_1 I_d, \ldots, \beta_N I_d)$, and some $\Psi, \Phi \subset \mathbb{R}^p$. Then the minimal residual of $\hat{f}(\lambda^{k+1}) + \hat{g}(\lambda^{k+1})$ is obtained by setting $\tau_i^k = 1/\sqrt{\alpha_i\,\beta_i}$, $\forall i = 1, \ldots, N$.

Proof. Substituting subgradients $\partial\hat{f}(\hat{\lambda}), \partial\hat{g}(\lambda)$ into the generalized DRS steps (43, 44), and using our linear assumption (46) yields

$$0 \in (T^k)^{-1}(\hat{\lambda}^{k+1} - \lambda^k) + (M_\alpha\,\hat{\lambda}^{k+1} + \Psi) + (M_\beta\,\lambda^k + \Phi)$$
$$0 \in (T^k)^{-1}(\lambda^{k+1} - \lambda^k) + (M_\alpha\,\hat{\lambda}^{k+1} + \Psi) + (M_\beta\,\lambda^{k+1} + \Phi).$$

Since T^k, M_α, M_β are diagonal matrices, we can split the equations into independent blocks, $\forall i = 1, \ldots, N$,

$$0 \in (\hat{\lambda}_i^{k+1} - \lambda_i^k)/\tau_i^k + (\alpha_i\,\hat{\lambda}^{k+1} + \Psi_i) + (\beta_i\,\lambda^k + \Phi_i)$$
$$0 \in (\lambda_i^{k+1} - \lambda_i^k)/\tau_i^k + (\alpha_i\,\hat{\lambda}^{k+1} + \Psi_i) + (\beta_i\,\lambda^{k+1} + \Phi_i).$$

Applying Proposition 1 in (Xu et al., 2017a) to each block, $\tau_i^k = 1/\sqrt{\alpha_i\,\beta_i}$ minimizes the block residual represented by $r_{DR,i}^{k+1} = \|(\alpha_i + \beta_i)\lambda^{k+1} + (a_i + b_i)\|$, where $a_i \in \Psi_i, b_i \in \Phi_i$. Hence the residual norm at step $k+1$, which is $\|(M_\alpha + M_\beta)\lambda^{k+1} + (a+b)\| = \sqrt{\sum_{i=1}^N (r_{DR,i}^{k+1})^2}$ is minimized by setting $\tau_i^k = 1/\sqrt{\alpha_i\,\beta_i}$, $\forall i = 1, \ldots, N$. □

5.3. Stepsize estimation for consensus problems

Thanks to the equivalence of ADMM and DRS, Proposition 1 can also be used to guide the selection of the "optimal" penalty parameter. We now show that the generalized spectral stepsizes can be estimated from the ADMM iterates for the primal consensus problem (2), without explicitly supplying the dual functions.

The subgradients of dual functions $\partial\hat{f}$, $\partial\hat{g}$ can be computed from the ADMM iterates using the identities derived from (8, 9),

$$Au^{k+1} - b \in \partial\hat{f}(\hat{\lambda}^{k+1}) \;\text{ and }\; Bv^{k+1} \in \partial\hat{g}(\lambda^{k+1}). \tag{47}$$

For the consensus problem we have $A = I_{dN}$, $B = -(I_d; \ldots; I_d)$, and $b = 0$, and so

$$(u_1^{k+1}; \ldots; u_N^{k+1}) \in \partial\hat{f}(\hat{\lambda}^{k+1}) \tag{48}$$

$$-(\underbrace{v^{k+1}; \ldots; v^{k+1}}_{N \text{ duplicates of } v^{k+1}}) \in \partial\hat{g}(\lambda^{k+1}). \tag{49}$$

If we approximate the behavior of these sub-gradients using the linear approximation (46), and break the subgradients into blocks (one for each worker node), we get (omitting iteration index k for clarity)

$$u_i = \alpha_i\,\hat{\lambda}_i + a_i \text{ and } -v = \beta_i\,\lambda_i + b_i,\ \forall i \tag{50}$$

where α_i and β_i represent the *curvature* of local functions $\hat{f}_i$ and $\hat{g}_i$ on the ith node.

We select stepsizes with a two step procedure, which follows the spectral stepsize literature. First, we estimate the local curvature parameters, α_i and β_i, by finding least-squares solutions to (50). Second, we plug these curvature estimates into the formula $\tau_i^k = 1/\sqrt{\alpha_i \beta_i}$. This formula produces the optimal stepsize when $\hat{f}$ and $\hat{g}$ are well approximated by a linear function, as shown in Proposition 1.

For notational convenience, we work with the quantities $\hat{\alpha}_i^k = 1/\alpha_i$, $\hat{\beta}_i^k = 1/\beta_i$, which are estimated on each node using the current iterates $u_i^k, v^k, \lambda_i^k, \hat{\lambda}_i^k$ and also an older iterate $u_i^{k_0}, v^{k_0}, \lambda_i^{k_0}, \hat{\lambda}_i^{k_0}, k_0 < k$. Defining $\Delta u_i^k = u_i^k - u_i^{k_0}$, $\Delta\hat{\lambda}_i^k = \hat{\lambda}_i^k - \hat{\lambda}_i^{k_0}$ and following the literature for Barzilai-Borwein/spectral stepsize estimation, there are two least squares estimators that can be obtained from (50):

$$\hat{\alpha}^k_{\text{SD},i} = \frac{\langle \Delta\hat{\lambda}_i^k, \Delta\hat{\lambda}_i^k\rangle}{\langle \Delta u_i^k, \Delta\hat{\lambda}_i^k\rangle} \text{ and } \hat{\alpha}^k_{\text{MG},i} = \frac{\langle \Delta u_i^k, \Delta\hat{\lambda}^k\rangle}{\langle \Delta u_i^k, \Delta u_i^k\rangle} \quad (51)$$

where SD stands for *steepest descent*, and MG stands for *minimum gradient*. (Zhou et al., 2006) recommend using a hybrid of these two estimators, and choosing

$$\hat{\alpha}_i^k = \begin{cases} \hat{\alpha}^k_{\text{MG},i} & \text{if } 2\,\hat{\alpha}^k_{\text{MG},i} > \hat{\alpha}^k_{\text{SD},i} \\ \hat{\alpha}^k_{\text{SD},i} - \hat{\alpha}^k_{\text{MG},i}/2 & \text{otherwise.} \end{cases} \quad (52)$$

It was observed that this choice worked well for non-distributed ADMM in (Xu et al., 2017a). We can similarly estimate $\hat{\beta}_i^k$ from $\Delta v^k = -v^k + v^{k_0}$ and $\Delta\lambda_i^k = \lambda_i^k - \lambda_i^{k_0}$.

ACADMM estimates the curvatures in the original d-dimensional feature space, and avoids estimating the curvature in the higher Nd-dimensional feature space (which grows with the number of nodes N in AADMM (Xu et al., 2017a)), which is especially useful for heterogeneous data with different distributions allocated to different nodes. The overhead of our adaptive scheme is only a few inner products, and the computation is naturally distributed on different workers.

5.4. Safeguarding and convergence

Spectral stepsizes for gradient descent methods are equipped with safeguarding strategies like backtracking line search to handle inaccurate curvature estimation and to guarantee convergence. To safeguard the proposed spectral penalty parameters, we check whether our linear subgradient assumption is reasonable before updating the stepsizes. We do this by testing that the correlations

$$\alpha^k_{\text{cor},i} = \frac{\langle \Delta u_i^k, \Delta\hat{\lambda}_i^k\rangle}{\|\Delta u_i^k\|\,\|\Delta\hat{\lambda}_i^k\|} \text{ and } \beta^k_{\text{cor},i} = \frac{\langle \Delta v^k, \Delta\lambda_i^k\rangle}{\|\Delta v^k\|\,\|\Delta\lambda_i^k\|}, \quad (53)$$

are bounded away from zero by a fixed threshold. We also bound changes in the penalty parameter by $(1 + C_{\text{cg}}/k^2)$ according to Assumption 1, which was shown in Theorem 1 and Theorem 2 to guarantee convergence. The final safeguarded ACADMM rule is

$$\hat{\tau}_i^{k+1} = \begin{cases} \sqrt{\hat{\alpha}_i^k \hat{\beta}_i^k} & \text{if } \alpha^k_{\text{cor},i} > \epsilon^{\text{cor}} \text{ and } \beta^k_{\text{cor},i} > \epsilon^{\text{cor}} \\ \hat{\alpha}_i^k & \text{if } \alpha^k_{\text{cor},i} > \epsilon^{\text{cor}} \text{ and } \beta^k_{\text{cor},i} \le \epsilon^{\text{cor}} \\ \hat{\beta}_i^k & \text{if } \alpha^k_{\text{cor},i} \le \epsilon^{\text{cor}} \text{ and } \beta^k_{\text{cor},i} > \epsilon^{\text{cor}} \\ \tau_i^k & \text{otherwise,} \end{cases} \quad (54)$$

$$\tau_i^{k+1} = \max\{\min\{\hat{\tau}_i^{k+1}, (1 + \frac{C_{\text{cg}}}{k^2})\tau_i^k\}, \frac{\tau_i^k}{1 + C_{\text{cg}}/k^2}\}.$$

Algorithm 1 Adaptive consensus ADMM (ACADMM)

Input: initialize v^0, λ_i^0, τ_i^0, $k_0 = 0$,
while not converge by (7) **and** $k <$ maxiter **do**
  Locally update u_i^k on each node by (3)
  Globally update v^k on central server by (4)
  Locally update dual variable λ_i^k on each node by (5)
  if $\text{mod}(k, T_f) = 1$ **then**
    Locally update $\hat{\lambda}_i^k = \lambda_i^{k-1} + \tau_i^k(v^{k-1} - u_i^k)$
    Locally compute spectral stepsizes $\hat{\alpha}_i^k, \hat{\beta}_i^k$
    Locally estimate correlations $\alpha^k_{\text{cor},i}$, $\beta^k_{\text{cor},i}$
    Locally update τ_i^{k+1} using (54)
    $k_0 \leftarrow k$
  else
    $\tau_i^{k+1} \leftarrow \tau_i^k$
  end if
  $k \leftarrow k + 1$
end while

The complete *adaptive consensus ADMM* is shown in Algorithm 1. We suggest updating the stepsize every $T_f = 2$ iterations, fixing the safeguarding threshold $\epsilon^{\text{cor}} = 0.2$, and choosing a large convergence constant $C_{\text{cg}} = 10^{10}$.

6. Experiments & Applications

We now study the performance of ACADMM on benchmark problems, and compare to other methods.

6.1. Applications

Our experiments use the following test problems that are commonly solved using consensus methods.

Linear regression with elastic net regularizer. We consider consensus formulations of the elastic net (Zou & Hastie, 2005) with f_i and g defined as,

$$f_i(u_i) = \frac{1}{2}\|D_i u_i - c_i\|^2,\ g(v) = \rho_1|v| + \frac{\rho_2}{2}\|v\|^2, \quad (55)$$

where $D_i \in \mathbb{R}^{n_i \times m}$ is the data matrix on node i, and c_i is a vector of measurements.

Sparse logistic regression with ℓ_1 regularizer can be written in the consensus form for distributed computing,

Table 1: Iterations (and runtime in seconds);128 cores are used; absence of convergence after n iterations is indicated as $n+$.

Application	Dataset	#samples × #features [1]	CADMM (Boyd et al., 2011)	RB-ADMM (He et al., 2000)	AADMM (Xu et al., 2017a)	CRB-ADMM (Song et al., 2016)	**Proposed ACADMM**
Elastic net regression	Synthetic1	64000 × 100	1000+(1.27e4)	94(1.22e3)	**43(563)**	106(1.36e3)	**48(623)**
	Synthetic2	64000 × 100	1000+(1.27e4)	130(1.69e3)	341(4.38e3)	140(1.79e3)	**57(738)**
	MNIST	60000 × 784	100+(1.49e4)	88(1.29e3)	40(5.99e3)	87(1.27e4)	**14(2.18e3)**
	CIFAR10 [2]	10000 × 3072	100+(1.04e3)	100+(1.06e3)	100+(1.05e3)	100+(1.05e3)	**35(376)**
	News20	19996 × 1355191	100+(4.61e3)	100+(4.60e3)	100+(5.17e3)	100+(4.60e3)	**78(3.54e3)**
	RCV1	20242 × 47236	33(1.06e3)	31(1.00e3)	20(666)	31(1.00e3)	**8(284)**
	Realsim	72309 × 20958	32(5.91e3)	30(5.59e3)	14(2.70e3)	30(5.57e3)	**9(1.80e3)**
Sparse logistic regression	Synthetic1	64000 × 100	138(137)	78(114)	80(101)	48(51.9)	**24(29.9)**
	Synthetic2	64000 × 100	317(314)	247(356)	1000+(1.25e3)	1000+(1.00e3)	**114(119)**
	MNIST	60000 × 784	325(444)	212(387)	325(516)	203(286)	**149(218)**
	CIFAR10	10000 × 3072	310(700)	152(402)	310(727)	149(368)	**44(118)**
	News20	19996 × 1355191	316(4.96e3)	211(3.84e3)	316(6.36e3)	207(3.73e3)	**137(2.71e3)**
	RCV1	20242 × 47236	155(115)	155(116)	155(137)	155(115)	**150(114)**
	Realsim	72309 × 20958	184(77)	184(77)	184(85)	183(77)	**159(68)**
Support Vector Machine	Synthetic1	64000 × 100	33(35.0)	33(49.8)	**19(27)**	26(28.4)	**21(25.3)**
	Synthetic2	64000 × 100	283(276)	69(112)	1000+(1.59e3)	81(97.4)	**25(39.0)**
	MNIST	60000 × 784	1000+(930)	172(287)	73(127)	285(340)	**41(88.0)**
	CIFAR10	10000 × 3072	1000+(774)	227(253)	231(249)	1000+(1.00e3)	**62(60.2)**
	News20	19996 × 1355191	259(2.63e3)	262(2.74e3)	259(3.83e3)	267(2.78e3)	**217(2.37e3)**
	RCV1	20242 × 47236	47(21.7)	47(21.6)	47(31.1)	40(19.0)	**27(15.4)**
	Realsim	72309 × 20958	1000+(76.8)	1000+(77.6)	442(74.4)	1000+(79.3)	**347(41.6)**
SDP	Ham-9-5-6	512 × 53760	100+(2.01e3)	100+(2.14e3)	35(860)	100+(2.14e3)	**30(703)**

[1] #vertices × #edges for SDP; [2]We only use the first training batch of CIFAR10.

$$f_i(u_i) = \sum_{j=1}^{n_i} \log(1 + \exp(-c_{i,j} D_{i,j}^T u_i)),\ g(v) = \rho|v| \quad (56)$$

where $D_{i,j} \in \mathbb{R}^m$ is the jth sample, and $c_{i,j} \in \{-1, 1\}$ is the corresponding label. The minimization sub-step (3) in this case is solved by L-BFGS (Liu & Nocedal, 1989).

Support Vector Machines (SVMs) minimize the distributed objective function (Goldstein et al., 2016)

$$f_i(u_i) = C \sum_{j=1}^{n_i} \max\{1 - c_{i,j} D_{i,j}^T u_i, 0\},\ g(v) = \frac{1}{2}\|v\|_2^2 \quad (57)$$

where $D_{i,j} \in \mathbb{R}^m$ is the jth sample on the ith node, and $c_{i,j} \in \{-1, 1\}$ is its label. The minimization (3) is solved by dual coordinate ascent (Chang & Lin, 2011).

Semidefinite programming (SDP) can be distributed as,

$$f_i(U_i) = \iota\{\mathcal{D}_i(U_i) = c_i\},\ g(v) = \langle F, V\rangle + \iota\{V \succeq 0\} \quad (58)$$

where $\iota\{S\}$ is a characteristic function that is 0 if condition S is satisfied and infinity otherwise. $V \succeq 0$ indicates that V is positive semidefinite. V, F, $D_{i,j} \in \mathbb{R}^{n\times n}$ are symmetric matrices, $\langle X, Y\rangle = \text{trace}(X^T Y)$ denotes the inner product of X and Y, and $\mathcal{D}_i(X) = (\langle D_{i,1}, X\rangle; \ldots; \langle D_{i,m_i}, X\rangle)$.

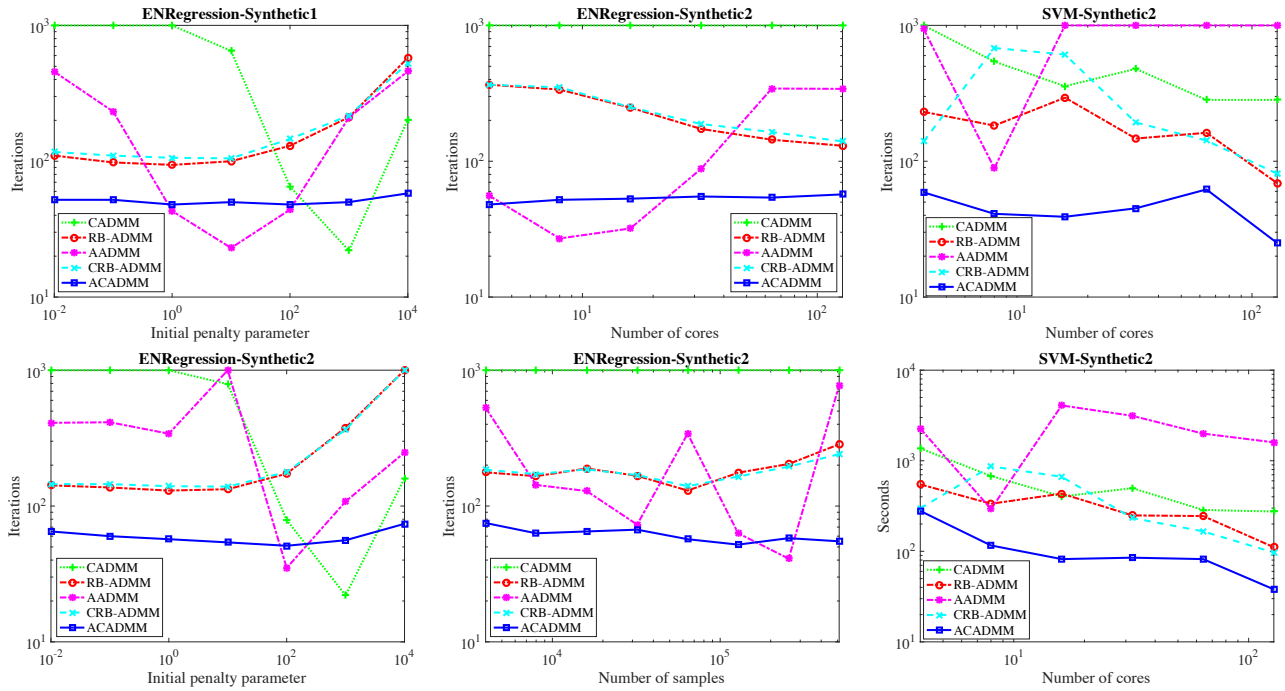
6.2. Experimental Setup

We test the problems in Section 6.1 with synthetic and real datasets. The number of samples and features are specified in Table 1. *Synthetic1* contains samples from a normal distribution, and *Synthetic2* contains samples from a mixture of 10 random Gaussians. *Synthetic2* is heterogeneous because the data block on each individual node is sampled from only 1 of the 10 Gaussians. We also acquire large empirical datasets from the LIBSVM webpage (Liu et al., 2009), as well as MNIST digital images (LeCun et al., 1998), and CIFAR10 object images (Krizhevsky & Hinton, 2009). For binary classification tasks (SVM and logreg), we equally split the 10 category labels of MNIST and CIFAR into "positive" and "negative" groups. We use a graph from the *Seventh DIMACS Implementation Challenge on Semidefinite and Related Optimization Problems* following (Burer & Monteiro, 2003) for Semidefinite Programming (SDP). The regularization parameter is fixed at $\rho = 10$ in all experiments.

Consensus ADMM (CADMM) (Boyd et al., 2011), residual balancing (RB-ADMM) (He et al., 2000), adaptive ADMM (AADMM) (Xu et al., 2017a), and consensus residual balancing (CRB-ADMM) (Song et al., 2016) are implemented and reported for comparison. Hyperparameters of these methods are set as suggested by their creators. The initial penalty is fixed at $\tau_0 = 1$ for all methods unless otherwise specified.

6.3. Convergence results

Table 1 reports the convergence speed in iterations and wall-clock time (secs) for various test cases. These experiments are performed with 128 cores on a Cray XC-30 supercomputer. CADMM with default penalty $\tau = 1$ (Boyd et al., 2011) is often slow to converge. ACADMM outperforms the other ADMM variants on all the real-world



(a) Sensitivity of iteration count to initial penalty τ_0. Synthetic problems of EN regression are studied with 128 cores.

(b) Sensitivity of iteration count to number of cores (top) and number of samples (bottom).

(c) Sensitivity of iteration count (top) and wall time (bottom) to number of cores.

Figure 1: ACADMM is robust to the initial penalty τ, number of cores N, and number of training samples.

datasets, and is competitive with AADMM on two homogeneous synthetic datasets where the curvature may be globally estimated with a scalar.

ACADMM is more reliable than AADMM since the curvature estimation becomes difficult for high dimensional variables. RB is relatively stable but sometimes has difficulty finding the exact optimal penalty, as the adaptation can stop because the difference of residuals are not significant enough to trigger changes. RB does not change the initial penalty in several experiments such as logistic regression on RCV1. CRB achieves comparable results with RB, which suggests that the relative sizes of local residuals may not always be very informative. ACADMM significantly boosts AADMM and the local curvature estimations are helpful in practice.

6.4. Robustness and sensitivity

Fig. 1a shows that the practical convergence of ADMM is sensitive to the choice of penalty parameter. ACADMM is robust to the selection of the initial penalty parameter and achieves promising results for both homogeneous and heterogeneous data, comparable to ADMM with a fine-tuned penalty parameter.

We study scalability of the method by varying the number of workers and training samples (Fig. 1b). ACADMM is fairly robust to the scaling factor. AADMM occasionally performs well when small numbers of nodes are used, while ACADMM is much more stable. RB and CRB are more stable than AADMM, but cannot compete with ACADMM. Fig. 1c (bottom) presents the acceleration in (wall-clock secs) achieved by increasing the number of workers.

Finally, ACADMM is insensitive to the safeguarding hyper-parameters, correlation threshold ϵ^{cor} and convergence constant C_{cg}. Though tuning these parameters may further improve the performance, the fixed default values generally perform well in our experiments and enable ACADMM to run without user oversight. In further experiments in the supplementary material, we also show that ACADMM is fairly insensitive to the regularization parameter ρ in our classification/regression models.

7. Conclusion

We propose ACADMM, a fully automated algorithm for distributed optimization. Numerical experiments on various applications and real-world datasets demonstrate the efficiency and robustness of ACADMM. We also prove a $O(1/k)$ convergence rate for ADMM with adaptive penalties under mild conditions. By automating the selection of algorithm parameters, adaptive methods make distributed systems more reliable, and more accessible to users that lack expertise in optimization.

Acknowledgements

ZX , GT, HL and TG 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. GT was partially supported by the DOD High Performance Computing Modernization Program. 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.

References

Banert, Sebastian, Bot, Radu Ioan, and Csetnek, Ernö Robert. Fixing and extending some recent results on the admm algorithm. *arXiv preprint arXiv:1612.05057*, 2016.

Boyd, Stephen, Parikh, Neal, Chu, Eric, Peleato, Borja, and Eckstein, Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. and Trends in Mach. Learning*, 3:1–122, 2011.

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

Chang, Chih-Chung and Lin, Chih-Jen. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

Chang, Tsung-Hui, Hong, Mingyi, Liao, Wei-Cheng, and Wang, Xiangfeng. Asynchronous distributed alternating direction method of multipliers: Algorithm and convergence analysis. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4781–4785. IEEE, 2016.

Davis, Damek and Yin, Wotao. Faster convergence rates of relaxed peaceman-rachford and admm under regularity assumptions. *arXiv preprint arXiv:1407.5210*, 2014.

Eckstein, Jonathan and Bertsekas, Dimitri. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, 1992.

França, Guilherme and Bento, José. An explicit rate bound for over-relaxed admm. In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pp. 2104–2108. IEEE, 2016.

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

Ghadimi, Euhanna, Teixeira, André, Shames, Iman, and Johansson, Mikael. Optimal parameter selection for the alternating direction method of multipliers: quadratic problems. *IEEE Trans. Autom. Control*, 60:644–658, 2015.

Giselsson, Pontus and Boyd, Stephen. Linear convergence and metric selection in douglas-rachford splitting and admm. 2016.

Glowinski, Roland and Marroco, A. 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: Modlisation Mathmatique et Analyse Numrique*, 9:41–76, 1975.

Goldfarb, Donald, Ma, Shiqian, and Scheinberg, Katya. Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming*, 141(1-2):349–382, 2013.

Goldstein, Tom and Setzer, Simon. High-order methods for basis pursuit. *UCLA CAM Report*, pp. 10–41, 2010.

Goldstein, Tom, O'Donoghue, Brendan, Setzer, Simon, and Baraniuk, Richard. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014.

Goldstein, Tom, Li, Min, and Yuan, Xiaoming. Adaptive primal-dual splitting methods for statistical learning and image processing. In *Advances in Neural Information Processing Systems*, pp. 2080–2088, 2015.

Goldstein, Tom, Taylor, Gavin, Barabin, Kawika, and Sayre, Kent. Unwrapping ADMM: efficient distributed computing via transpose reduction. In *AISTATS*, 2016.

He, Bingsheng and Yuan, Xiaoming. On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM Journal on Numerical Analysis*, 50(2): 700–709, 2012.

He, Bingsheng and Yuan, Xiaoming. On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers. *Numerische Mathematik*, 130: 567–577, 2015.

He, Bingsheng, Yang, Hai, and Wang, Shengli. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Jour. Optim. Theory and Appl.*, 106(2):337–356, 2000.

Kadkhodaie, Mojtaba, Christakopoulou, Konstantina, Sanjabi, Maziar, and Banerjee, Arindam. Accelerated alternating direction method of multipliers. In *Proceedings of the 21th ACM SIGKDD*, pp. 497–506, 2015.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. 2009.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lin, Zhouchen, Liu, Risheng, and Su, Zhixun. Linearized alternating direction method with adaptive penalty for low-rank representation. In *NIPS*, pp. 612–620, 2011.

Liu, Dong C and Nocedal, Jorge. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Liu, Jun, Chen, Jianhui, and Ye, Jieping. Large-scale sparse logistic regression. In *ACM SIGKDD*, pp. 547–556, 2009.

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

Ouyang, Hua, He, Niao, Tran, Long, and Gray, Alexander G. Stochastic alternating direction method of multipliers. *ICML (1)*, 28:80–88, 2013.

Raghunathan, Arvind and Di Cairano, Stefano. Alternating direction method of multipliers for strictly convex quadratic programs: Optimal parameter selection. In *American Control Conf.*, pp. 4324–4329, 2014.

Rockafellar, R. *Convex Analysis*. Princeton University Press, 1970.

Song, Changkyu, Yoon, Sejong, and Pavlovic, Vladimir. Fast ADMM algorithm for distributed optimization with adaptive penalty. *AAAI*, 2016.

Studer, Christoph, Goldstein, Tom, Yin, Wotao, and Baraniuk, Richard G. Democratic representations. *arXiv preprint arXiv:1401.3420*, 2014.

Taylor, Gavin, Burmeister, Ryan, Xu, Zheng, Singh, Bharat, Patel, Ankit, and Goldstein, Tom. Training neural networks without gradients: A scalable ADMM approach. *ICML*, 2016.

Tian, Wenyi and Yuan, Xiaoming. Faster alternating direction method of multipliers with a worst-case o ($1/n^2$) convergence rate. 2016.

Xu, Zheng, Figueiredo, Mario AT, and Goldstein, Thomas. Adaptive ADMM with spectral penalty parameter selection. *AISTATS*, 2017a.

Xu, Zheng, Figueiredo, Mario AT, Yuan, Xiaoming, Studer, Christoph, and Goldstein, Thomas. Adaptive relaxed ADMM: Convergence theory and practical implementation. *CVPR*, 2017b.

Zhang, Ruiliang and Kwok, James T. Asynchronous distributed ADMM for consensus optimization. In *ICML*, pp. 1701–1709, 2014.

Zhou, Bin, Gao, Li, and Dai, Yu-Hong. Gradient methods with adaptive step-sizes. *Computational Optimization and Applications*, 35:69–86, 2006.

Zou, Hui and Hastie, Trevor. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2): 301–320, 2005.