On the $O(1/k)$ Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers*

Ermin Wei† and Asuman Ozdaglar†

***Abstract*— We consider a network of agents that are cooperatively solving a global optimization problem, where the objective function is the sum of privately known local objective functions of the agents and the decision variables are coupled via linear constraints. Recent literature focused on special cases of this formulation and studied their distributed solution through either subgradient based methods with $O(1/\sqrt{k})$ rate of convergence (where k is the iteration number) or Alternating Direction Method of Multipliers (ADMM) based methods, which require a synchronous implementation and a globally known order on the agents. In this paper, we present a novel asynchronous ADMM based distributed method for the general formulation and show that it converges at the rate $O(1/k)$.**

I. Introduction

We consider the following optimization problem with a separable objective function and linear constraints:

$$\min_{x_i\in X_i, z\in Z} \quad \sum_{i=1}^{N} f_i(x_i) \qquad s.t. \quad Dx+Hz=0. \tag{1}$$

Here each $f_i:\mathbb{R}^n\to\mathbb{R}$ is a (possibly nonsmooth) convex function, X_i and Z are closed convex subsets of $\mathbb{R}^n$ and $\mathbb{R}^W$, and D and H are matrices of dimensions $W\times nN$ and $W\times W$. The decision variable x is given by the partition $x=[x_1',\ldots,x_N']'\in\mathbb{R}^{nN}$. We denote by set X the product of sets X_i, hence the constraint on x can be written compactly as $x\in X$.

Our focus on this formulation is motivated by *distributed multi-agent optimization problems*, which attracted much recent attention in the optimization, control and signal processing communities. Such problems involve resource allocation, information processing, and learning among a set $\{1,\ldots,N\}$ of distributed agents connected through a network $G=(V,E)$, where E denotes the set of M undirected edges between the agents. In such applications, each agent has access to a privately known local objective (or cost) function, which represents the negative utility or the loss agent i incurs at the decision variable x. The goal is to collectively solve a global optimization problem

$$\min \sum_{i=1}^{N} f_i(x) \qquad s.t. \quad x\in X. \tag{2}$$

This problem can be reformulated in the general formulation (1) by introducing a local copy x_i for each node i and imposing the constraint $x_i=x_j$ for all agents i and j with edge $(i,j)\in E$. Since these problems often lack a centralized processing unit, it is imperative that iterative solutions of problem (2) involve decentralized computations.

Though there have been many important advances in the design of decentralized optimization algorithms for multi-agent optimization problems, several challenges still remain. First, many of these algorithms are based on first-order subgradient methods [1], [12], [9], which for general convex problems have slow convergence rates (given by $O(1/\sqrt{k})$ where k is the iteration number),[1] making them impractical in many large scale applications. Second, with the exception of [6] and [10], existing algorithms are synchronous, meaning that computations are simultaneously performed according to some global clock, but this often goes against the highly decentralized nature of the problem, which precludes such global information being available to all nodes. Moreover, neither of the works [6] and [10] provides a rate of convergence analysis of the asynchronous algorithm.

In this paper, we focus on the more general formulation (1) and propose an asynchronous decentralized algorithm based on the classical Alternating Direction Method of Multipliers (ADMM) (see [2], [3], [4], [5], [13] for convergence analysis of classical ADMM, and also [8], [11] for its applications and analysis in distributed network settings), where we adopt the following asynchronous implementation: at each iteration k, a random subset of the constraints is selected, which in turn selects the components of x that appear in these constraints. We refer to the selected constraints as *active constraints* and selected components as the *active components (or agents)*. We design an ADMM-type algorithm which at each iteration updates the primal and dual variables using information from the active parts of the problem. Under the assumption that each constraint has a positive probability of being selected, we establish that the (primal) asynchronous iterates generated by this algorithm converge almost surely to an optimal solution. Under further assumption of compactness of the constraint sets X and Z, we provide a *performance guarantee of* $O(1/k)$, which to our knowledge is the best rate available for this problem without additional smoothness

*This work was supported by National Science Foundation under Career grant DMI-0545910, AFOSR MURI FA9550-09-1-0538, and ONR Basic Research Challenge No. N000141210997.

†Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology

[1]Under further assumption of Lipschitz and bounded gradient, gradient methods can converge at rate $O(1/k^2)$, see [7].

assumptions.

The paper is organized as follows: in Section II, we focus on the more general formulation (1), present the asynchronous ADMM algorithm. Section III contains our convergence and rate of convergence analysis. Section IV concludes with closing remarks. Due to space limitation, we omit the proofs here. We refer the reader to [14] for the missing details.

Basic Notation and Notions:

A vector is viewed as a column vector. For a matrix A, we write $[A]_i$ to denote the i^{th} column of matrix A, and $[A]^j$ to denote the j^{th} row of matrix A. For a vector x, x_i denotes the i^{th} component of the vector. We use x' and A' to denote the transpose of a vector x and a matrix A respectively. We use standard Euclidean norm (i.e., 2-norm) unless otherwise noted, i.e., for a vector x in $\mathbb{R}^n$, $||x|| = \left(\sum_{i=1}^n x_i^2\right)^{\frac{1}{2}}$.

II. Asynchronous ADMM Algorithm

We present the problem formulation and assumptions in Section II-A. In Section II-B, we discuss the asynchronous implementation considered in the rest of this paper that involves updating a subset of components of the decision vector at each time using partial information about problem data and without need for a global coordinator. Section II-C contains the details of the asynchronous ADMM algorithm.

A. Problem Formulation and Assumptions

We consider the optimization problem given in (1), which arises in large-scale multi-agent (or processor) environments where problem data is distributed across N agents, i.e., each agent i has access only to the component function f_i and maintains the decision variable component x_i. The constraints usually represent the coupling across components of the decision variable imposed by the underlying connectivity among the agents. Motivated by such applications, we will refer to each component function f_i as the *local objective function* and use the notation $F : \mathbb{R}^{nN} \to \mathbb{R}$ to denote the *global objective function* given by their sum $F(x) = \sum_{i=1}^N f_i(x_i)$.

We adopt the following standard assumption.

Assumption 1: (Existence of a Saddle Point) The Lagrangian function of problem (1), $L(x, z, p) = F(x) - p'(Dx + Hz)$, has a saddle point, i.e., there exists a solution-multiplier pair (x^*, z^*, p^*) with $L(x^*, z^*, p) \leq L(x^*, z^*, p^*) \leq L(x, z, p^*)$ for all x in X, z in Z and p in $\mathbb{R}^W$.

Moreover, we assume that the matrices have special structure that enables solving problem (1) in an asynchronous manner:

Assumption 2: (*Decoupled Constraints*) Matrix H is diagonal and invertible. Each row of matrix D has exactly one nonzero element and matrix D has no columns of all zeros.[2]

The diagonal structure of matrix H implies that each component of vector z appears in exactly one linear constraint. The conditions that each row of matrix D has only one nonzero element and matrix D has no column of zeros guarantee the columns of matrix D are linearly independent and hence matrix $D'D$ is positive definite. The condition on matrix D implies that each row of the constraint $Dx+Hz = 0$ involves exactly one x_i. This assumption is satisfied by an equivalent transformation of the distributed multi-agent optimization problem that motivates this work.

B. Asynchronous Algorithm Implementation

In the large scale multi-agent applications described above, it is essential that the iterative solution of the problem involves computations performed by agents in a decentralized manner (with access to local information) with as little coordination as possible. This necessitates an asynchronous implementation in which some of the agents become active (randomly) in time and update the relevant components of the decision variable using partial and local information about problem data while keeping the rest of the components of the decision variable unchanged. This removes the need for a centralized coordinator or global clock, which is an unrealistic requirement in such decentralized environments.

To describe the asynchronous algorithm implementation we consider in this paper more formally, we first introduce some notation. We call a partition of the set $\{1, \ldots, W\}$ a *proper partition* if it has the property that if z_i and z_j are coupled in the constraint set Z, i.e., value of z_i affects the constraint on z_j for any z in set Z, then i and j belong to the same partition, i.e., $\{i, j\} \subset \psi$ for some ψ in the partition. We let Π be a proper partition of the set $\{1, \ldots, W\}$, which forms a partition of the set of W rows of the linear constraint $Dx+Hz = 0$. For each ψ in Π, we define $\Phi(\psi)$ to be the set of indices i, where x_i appears in the linear constraints in set ψ. Note that $\Phi(\psi)$ is an element of the power set $2^{\{1,\ldots,N\}}$.

At each iteration of the asynchronous algorithm, two random variables Φ^k and Ψ^k are realized. While the pair (Φ^k, Ψ^k) is correlated for each iteration k, these variables are assumed to be independent and identically distributed across iterations. At each iteration k, first the random variable Ψ^k is realized. The realized value, denoted by ψ^k, is an element of the proper partition Π and selects a subset of the linear constraints $Dx+Hz = 0$. The random variable Φ^k then takes the realized value $\phi^k = \Phi(\psi^k)$. We can view this process as activating a subset of the coupling constraints and the components that are involved in these constraints. If $l \in \psi^k$, we say constraint l as well as its associated dual variable p_l is *active* at iteration k. Moreover, if $i \in \Phi(\psi^k)$, we say

[2]We assume without loss of generality that each x_i is involved at least in one of the constraints, otherwise, we could remove it from the problem and optimize it separately. Similarly, the diagonal elements of matrix H are assumed to be non-zero, otherwise, that component of variable z can be dropped from the optimization problem.

that component i or agent i is *active* at iteration k. We use the notation $\bar{\phi}^k$ to denote the complement of set ϕ^k in set $\{1,\dots,N\}$ and similarly $\bar{\psi}^k$ to denote the complement of set ψ^k in set $\{1,\dots,W\}$.

Our goal is to design an algorithm in which at each iteration k, only active components of the decision variable and active dual variables are updated using local cost functions of active agents and active constraints. To that end, we define $f^k : \mathbb{R}^{nN} \to \mathbb{R}$ as the sum of the local objective functions whose indices are in the subset ϕ^k, $f^k(x) = \sum_{i\in\phi^k} f_i(x_i)$. We denote by D_i the matrix in $\mathbb{R}^{W\times nN}$ that picks up the columns corresponding to x_i from matrix D and has zeros elsewhere. Similarly, we denote by H_l the diagonal matrix in $\mathbb{R}^{W\times W}$ which picks up the element in the l^{th} diagonal position from matrix H and has zeros elsewhere. Using this notation, we define the matrices $D_{\phi^k} = \sum_{i\in\phi^k} D_i$, and $H_{\psi^k} = \sum_{l\in\psi^k} H_l$.

We impose the following condition on the asynchronous algorithm.

Assumption 3: (*Infinitely Often Update) For all k and all ψ in the proper partition* Π, $\mathbb{P}(\Psi^k = \psi) > 0$.

This assumption ensures that each element of the partition Π is active infinitely often with probability 1. Since matrix D has no columns of all zeros, each of the x_i is involved in some constraints, and hence $\cup_{\psi\in\Pi}\Phi(\psi) = \{1,\dots,N\}$. The preceding assumption therefore implies that each agent i belongs to at least one set $\Phi(\psi)$ and therefore is active infinitely often with probability 1. From definition of the partition Π, we have $\cup_{\psi\in\Pi}\psi = \{1,\dots,W\}$. Thus, each constraint l is active infinitely often with probability 1.

C. Asynchronous ADMM Algorithm

We next describe the asynchronous ADMM algorithm for solving problem (1).

I. Asynchronous ADMM algorithm:

A Initialization: choose some arbitrary x^0 in X, z^0 in Z and $p^0 = 0$.

B At iteration k, random variables Φ^k and Ψ^k takes realizations ϕ^k and ψ^k. Function f^k and matrices D_{ϕ^k}, H_{ψ^k} are generated accordingly.

a The primal variable x is updated as $x^{k+1} \in \operatorname{argmin}_{x\in X} f^k(x) - (p^k)'D_{\phi^k}x + \frac{\beta}{2}\left|\left|D_{\phi^k}x + Hz^k\right|\right|^2$, with $x_i^{k+1} = x_i^k$, for i in $\bar{\phi}^k$.

b The primal variable z is updated as $z^{k+1} \in \operatorname{argmin}_{z\in Z} -(p^k)'H_{\psi^k}z + \frac{\beta}{2}\left|\left|H_{\psi^k}z + D_{\phi^k}x^{k+1}\right|\right|^2$, with $z_i^{k+1} = z_i^k$, for i in $\bar{\psi}^k$.

c The dual variable p is updated as

$$p^{k+1} = p^k - \beta[D_{\phi^k}x^{k+1} + H_{\psi^k}z^{k+1}]_{\psi^k}.$$

We assume that the minimizers in steps B.a and B.b exist, but need not be unique. This algorithm can be applied to solve problem (2) in a distributed way using random local clocks associated with edges to activate the components and constraints (see [14] for details).

III. Convergence Analysis for Asynchronous ADMM Algorithm

In this section, we study the convergence behavior of the asynchronous ADMM algorithm under Assumptions 1-3. Our proof relies on relating the asynchronous iterates to *full-information* iterates that would be generated by the algorithm that use full information about the cost functions and constraints at each iteration. We also introduce a weighted norm and weighted Lagrangian function where the weights are defined in terms of the probability distributions of random variables Ψ^k and Φ^k representing the active constraints and components. We use the weighted norm and properties of the full information iterates to construct a nonnegative supermartingale along the sequence $\{x^k, z^k, p^k\}$ generated by the asynchronous ADMM algorithm and use it to establish the almost sure convergence of this sequence to a saddle point of the Lagrangian function of problem (1). By relating the iterates generated by the asynchronous ADMM algorithm to the full information iterates through taking expectations of the weighted Lagrangian function, we can show that under a compactness assumption on the constraint sets X and Z, the asynchronous ADMM algorithm converges with rate $O(1/k)$ in expectation in terms of both objective function value and constraint violation.

We use the notation α_i to denote the probability that component x_i is active at one iteration, i.e., $\alpha_i = \mathbb{P}(i \in \Phi^k)$, and the notation λ_l to denote the probability that constraint l is active at one iteration, i.e., $\lambda_l = \mathbb{P}(l \in \Psi^k)$. Note that, since the random variables Φ^k (and Ψ^k) are independent and identically distributed for all k, these probabilities are the same across all iterations. We define a diagonal matrix Λ in $\mathbb{R}^{W\times W}$ with elements λ_l on the diagonal, i.e., $\Lambda_{ll} = \lambda_l$, for each $l \in \{1,\dots,W\}$. Since each constraint is assumed to be active with strictly positive probability [cf. Assumption 3], matrix Λ is positive definite. We write $\bar{\Lambda}$ to indicate the inverse of matrix Λ. Matrix $\bar{\Lambda}$ induces a *weighted vector norm* for p in $\mathbb{R}^W$ as $||p||^2_{\bar{\Lambda}} = p'\bar{\Lambda}p$. We define a *weighted Lagrangian function* $\tilde{L}(x,z,\mu) : \mathbb{R}^{nN}\times\mathbb{R}^W\times\mathbb{R}^W \to \mathbb{R}$ as

$$\tilde{L}(x,z,\mu) = \sum_{i=1}^{N}\frac{1}{\alpha_i}f_i(x_i) - \mu'\left(\sum_{i=1}^{N}\frac{1}{\alpha_i}D_i x + \sum_{l=1}\frac{1}{\lambda_l}H_l z\right). \tag{3}$$

Theorem 3.1: Let $\{x^k, z^k, p^k\}$ be the sequence generated by the asynchronous ADMM algorithm. The sequence $\{x^k, z^k, p^k\}$ converges almost surely to a saddle point of the Lagrangian function of problem (1).

We next analyze convergence rate of the asynchronous ADMM algorithm. The rate analysis is done with respect to

the time ergodic averages defined as $\bar{x}(T)$ in $\mathbb{R}^{nN}$, the time average of x^k up to and including iteration T, i.e.,

$$\bar{x}_i(T) = \frac{\sum_{1=1}^{T} x_i^k}{T}, \tag{4}$$

for all $i = 1, \ldots, N$,[3] and $\bar{z}(k)$ in $\mathbb{R}^W$ as

$$\bar{z}_l(T) = \frac{\sum_{k=1}^{T} z_l^k}{T}, \tag{5}$$

for all $l = 1, \ldots, W$.

We introduce some scalars $Q(\mu)$, $\bar{Q}$, $\bar{\theta}$ and $\tilde{L}^0$, all of which will be used to provide an upper bound on the constant term that appears in the rate analysis. Scalar $Q(\mu)$ is defined by $Q(\mu) = \max_{x\in X, z\in Z} -\tilde{L}(x, z, \mu)$, which implies $Q(\mu) \geq -\tilde{L}(x^{k+1}, z^{k+1}, \mu)$ for any realization of Ψ^k and Φ^k. For the rest of the section, we adopt the following assumption, which will be used to guarantee that scalar $Q(\mu)$ is well defined and finite:

Assumption 4: The sets X and Z are both compact.

Since the weighted Lagrangian function $\tilde{L}$ is continuous in x and z [cf. Eq. (3)], and all iterates (x^k, z^k) are in the compact set $X \times Z$, by Weierstrass theorem the maximization in the preceding equality is attained and finite.

Since function $\tilde{L}$ is linear in μ, the function $Q(\mu)$ is the maximum of linear functions and is thus convex and continuous in μ. We define scalar $\bar{Q}$ as $\bar{Q} = \max_{\mu=p^*-\alpha, ||\alpha||\leq 1} Q(\mu)$. The reason that such scalar $\bar{Q} < \infty$ exists is once again by Weierstrass theorem (maximization over a compact set).

We define vector $\bar{\theta}$ in $\mathbb{R}^W$ as $\bar{\theta} = p^* - \text{argmax}_{||u||\leq 1} \left|\left|p^0 - (p^* - u)\right|\right|_{\tilde{\Lambda}}^2$, such maximizer exists due to Weierstrass theorem and the fact that the set $||u|| \leq 1$ is compact and the function $\left|\left|p^0 - (p^* - u)\right|\right|_{\tilde{\Lambda}}^2$ is continuous. Scalar $\tilde{L}^0$ is defined by $\tilde{L}^0 = \max_{\theta=p^*-\alpha, ||\alpha||\leq 1} \tilde{L}(x^0, z^0, \theta)$. This scalar is well defined because the constraint set is compact and the function $\tilde{L}$ is continuous in θ.

Theorem 3.2: Let $\{x^k, z^k, p^k\}$ be the sequence generated by the asynchronous ADMM algorithm and (x^*, z^*, p^*) be a saddle point of the Lagrangian function of problem (1). Let the vectors $\bar{x}(T)$, $\bar{z}(T)$ be defined as in Eqs. (4) and (5), the scalars $\bar{Q}$, $\bar{\theta}$ and $\tilde{L}^0$ be defined as above and the function $\tilde{L}$ be defined as in Eq. (3). Then the following relations hold:$||\mathbb{E}(D\bar{x}(T) + H\bar{z}(T))|| \leq \frac{1}{T}\left[\bar{Q} + \tilde{L}^0 + \frac{1}{2\beta}\left|\left|p^0 - \bar{\theta}\right|\right|_{\tilde{\Lambda}}^2 + \frac{\beta}{2}\left|\left|H(z^0 - z^*)\right|\right|_{\tilde{\Lambda}}^2\right]$,
$||\mathbb{E}(F(\bar{x}(T))) - F(x^*)|| \leq \frac{||p^*||_\infty}{T}\left[\bar{Q} + \tilde{L}^0 + \frac{1}{2\beta}\left|\left|p^0 - p^*\right|\right|_{\tilde{\Lambda}}^2 + \frac{\beta}{2}\left|\left|H(z^0 - z^*)\right|\right|_{\tilde{\Lambda}}^2\right] + \frac{1}{T}\left[Q(p^*) + \tilde{L}(x^0, z^0, p^*) + \frac{1}{2\beta}\left|\left|p^0 - \bar{\theta}\right|\right|_{\tilde{\Lambda}}^2 + \frac{\beta}{2}\left|\left|H(z^0 - z^*)\right|\right|_{\tilde{\Lambda}}^2\right]$.

We remark that by Jensen's inequality and convexity of the function F, we have $F(\mathbb{E}(\bar{x}(T))) \leq \mathbb{E}(F(\bar{x}(T)))$, and the preceding results also holds true when we replace $\mathbb{E}(F(\bar{x}(T)))$ by $F(\mathbb{E}(\bar{x}(T)))$.

[3]Here the notation $\bar{x}_i(T)$ denotes the vector of length n corresponding to agent i.

IV. Conclusions

We developed a fully asynchronous ADMM based algorithm for a convex optimization problem with separable objective function and linear constraints. This problem is motivated by distributed multi-agent optimization problems where a (static) network of agents each with access to a privately known local objective function seek to optimize the sum of these functions using computations based on local information and communication with neighbors. We show that this algorithm converges almost surely to an optimal solution. Moreover, the rate of convergence of the objective function values and feasibility violation is given by $O(1/k)$. Future work includes investigating network effects (e.g., effects of communication noise, quantization) and time-varying network topology on the performance of the algorithm.

References

[1] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, MA, 1997.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[3] J. Eckstein. Augmented Lagrangian and Alternating Direction Methods for Convex Optimization: A Tutorial and Some Illustrative Computational Results Augmented Lagrangian and Alternating Direction Methods for Convex Optimization : A Tutorial and Some Il. *Rutcor Research Report*, 2012.

[4] 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.

[5] M. Hong and Z. Luo. On the Linear Convergence of the Alternating Direction Method of Multipliers. *Arxiv preprint*, 2012.

[6] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem. Asynchronous Distributed Optimization using a Randomized Alternating Direction Method of Multipliers. *Submitted to IEEE Conference on Decision and Control (CDC)*, 2013.

[7] D. Jakovetic, J. Xavier, and J. M. F. Moura. Fast Distributed Gradient Methods. *Arxiv preprint*, 2011.

[8] J. Mota, J. Xavier, P. Aguiar, and M. Püschel. D-ADMM : A Communication-Efficient Distributed Algorithm For Separable Optimization. *IEEE Transactions on Signal Processing*, 61(10):2718–2723, 2013.

[9] A. Nedich and A. Ozdaglar. Distributed Subgradient Methods for Multi-agent Optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[10] S. S. Ram, A. Nedich, and V. V. Veeravalli. Asynchronous Gossip Algorithms for Stochastic Optimization. *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2009.

[11] I. D. Schizas, R. Ribeiro, and G. B. Giannakis. Consensus in Ad Hoc WSNs with Noisy Links - Part I: Distributed Estimation of Deterministic Signals. *IEEE Transactions on Singal Processing*, 56:350–364, 2008.

[12] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1984.

[13] E. Wei and A. Ozdaglar. Distributed Alternating Direction Method of Multipliers. *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2012.

[14] E. Wei and A. Ozdaglar. On the $O(1/k)$ Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers. *LIDS report 2906, submitted to Mathematical Programming*, 2013.