On the $O(1/k)$ convergence of asynchronous distributed alternating Direction Method of Multipliers

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://dx.doi.org/10.1109/GlobalSIP.2013.6736937">http://dx.doi.org/10.1109/GlobalSIP.2013.6736937</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>Institute of Electrical and Electronics Engineers (IEEE)</td>
</tr>
<tr>
<td>Version</td>
<td>Author's final manuscript</td>
</tr>
<tr>
<td>Accessed</td>
<td>Fri Dec 28 16:49:58 EST 2018</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/90493">http://hdl.handle.net/1721.1/90493</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Creative Commons Attribution-Noncommercial-Share Alike</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td><a href="http://creativecommons.org/licenses/by-nc-sa/4.0/">http://creativecommons.org/licenses/by-nc-sa/4.0/</a></td>
</tr>
</tbody>
</table>
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 co-operatively 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:

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

Here each $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ 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 $x = [x^1, \ldots, x^N]^T \in \mathbb{R}^{nN}$. 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

$$
\min_{x \in X} \sum_{i=1}^{N} f_i(x) \quad \text{s.t.} \quad x \in X.
$$

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) 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 FA9550-09-1-0338, and ONR Basic Research Challenge No. N000141210997.

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

††Under further assumption of Lipschitz and bounded gradient, gradient methods can converge at rate $O(1/k^2)$, see [7].
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 $\mathbb{R}^n$, $||x|| = (\sum_{i=1}^{n} x_i^2)^{\frac{1}{2}}$.

II. ASYNCHRONOUS ADMM ALGORITHM

We present the problem formulation and assumptions in Section II-A. In Section II-B, we discuss the asynchronous algorithm implementation in which some of the agents become active (randomly) in time and update the relevant components of the decision variable using partial 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 $\psi$ in the partition. We let $\Pi$ 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 $\psi$ in $\Pi$, we define $\Phi(\psi)$ to be the set of indices $i$, where $x_i$ appears in the linear constraints in set $\psi$. 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 $\Phi^k$ and $\Psi^k$ are realized. While the pair $(\Phi^k, \Psi^k)$ is correlated for each iteration $k$, these variables are assumed to be independent and identically distributed as iterations. At each iteration $k$, first the random variable $\Psi^k$ is realized. The realized value, denoted by $\psi^k$, is an element of the proper partition $\Pi$ and selects a subset of the linear constraints $Dx + Hz = 0$. The random variable $\Phi^k$ 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

2We 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 \(\phi^k\) to denote the complement of set \(\phi^k\) in set \(\{1, \ldots, N\}\) and similarly \(\psi^k\) to denote the complement of set \(\psi^k\) in set \(\{1, \ldots, W\}\).

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

\[
\Psi^k = \sum_{i=1}^N \phi^k_i f_i(x_i)
\]

We impose the following condition on the asynchronous ADMM 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 \(\Psi^k\) and \(\Phi^k\) 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 \(\alpha_i\) to denote the probability that component \(x_i\) is active at one iteration, i.e., \(\alpha_i = P(i \in \phi^k)\), and the notation \(\lambda_l\) to denote the probability that constraint \(l\) is active at one iteration, i.e., \(\lambda_l = P(l \in \psi^k)\). Note that, since the random variables \(\Phi^k\) (and \(\Psi^k\)) are independent and identically distributed for all \(k\), these probabilities are the same across all iterations. We define a diagonal matrix \(\Lambda\) in \(\mathbb{R}^{W \times W}\) with elements \(\lambda_l\) on the diagonal, and \(\Lambda_{ll} = \lambda_l\) for each \(l \in \{1, \ldots, W\}\). Since each constraint is assumed to be active with strictly positive probability [cf. Assumption 3], matrix \(\Lambda\) is positive definite. We write \(\bar{\Lambda}\) to indicate the inverse of matrix \(\Lambda\). Matrix \(\bar{\Lambda}\) induces a weighted vector norm for \(p\) in \(\mathbb{R}^W\) as \(\|p\|_{\bar{\Lambda}}^2 = p^T \Lambda p\). 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

\[
\tilde{L}(x, z, \mu) = \sum_{i=1}^N \frac{1}{\alpha_i} f_i(x_i) - \mu^T \left( \sum_{i=1}^N \frac{1}{\alpha_i} D_i x_i + \sum_{l=1}^W \frac{1}{\lambda_l} H_l z_l \right)
\]

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

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 \( \tilde{z}(T) \) in \( \mathbb{R}^{nN} \), the time average of \( x^k \) up to and including iteration \( T \), i.e.,

\[
\tilde{z}_i(T) = \frac{\sum_{k=1}^{T} x^k}{T},
\]

for all \( i = 1, \ldots, N \) and \( \tilde{z}(k) \) in \( \mathbb{R}^W \) as

\[
\tilde{z}_i(T) = \frac{\sum_{k=1}^{T} z^k}{T},
\]

for all \( l = 1, \ldots, W \).

We introduce some scalars \( Q(\mu), \tilde{Q}, \tilde{\theta} \) 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 \( \Psi^k \) and \( \Phi^k \). 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. (4)], 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 \( \mu \), the function \( Q(\mu) \) is the maximum of linear functions and is thus convex and continuous in \( \mu \). We define scalar \( Q \) as \( Q = \max_{\mu} = \tilde{Q} \cdot \alpha |\alpha| \leq 1 \). The reason that such scalar \( Q \) is finite is once again by Weierstrass theorem (maximization over a compact set).

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

**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 \( \tilde{L}^0 \). Let the vectors \( \tilde{x}(T) \), \( \tilde{z}(T) \) be defined as in Eqs. \( \tilde{Q} \) and \( \tilde{L}^0 \), the scalars \( \tilde{Q} \), \( \tilde{\theta} \) and \( \tilde{L}^0 \) be defined as above and the function \( \tilde{L} \) be defined as in Eq. (5). Then the following relations hold:

\[
\left\| E(D\tilde{x}(T) + \tilde{L}(T)) \right\| \leq \frac{1}{T} \left[ \tilde{Q} + \tilde{L}^0 + \frac{1}{T} \left\| p^0 - \tilde{\theta} \right\|_A^2 + \frac{2}{T} \left\| H(z^0 - z^*) \right\|_A^2 \right],
\]

and the preceding results also holds true when we replace \( E(F(\tilde{x}(T))) \) by \( F(E(\tilde{x}(T))) \).

**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**


