Automatica ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Technical communique

Is ADMM always faster than Average Consensus?☆

Nicoletta Bof, Ruggero Carli *, Luca Schenato

Department of Information Engineering, University of Padova, via Gradenigo 6/a, 35131, Padova, Italy

ARTICLE INFO

Article history:
Received 28 March 2017
Received in revised form 6 September 2017
Accepted 27 November 2017
Available online xxxx

Keywords:
Distributed control
Optimization problems
Average Consensus
ADMM

ABSTRACT

There is a common belief that the ADMM, a popular algorithm employed for distributed convex optimization over graphs, is faster than another distributed algorithm typically referred as the Average Consensus. This belief is based on the observation that the ratio of the number of iterations necessary to achieve a desired error with respect to the optimal solution of the ADMM vs the Average Consensus goes to zero as the graph becomes larger or less connected. In this work, we provide a closed form expression for the rate of ADMM as a function of the essential spectral radius of the graph, which is a measure of connectivity of the graph, for scalar quadratic cost functions with identical curvature, and we show that its rate of convergence can be slower than the Average Consensus when the graph is highly connected. Moreover, via extensive simulations, we show that ADMM performance, differently from the average consensus, rapidly degrade as the cost functions become skewed, thus making the latter approach competitive also for sparse graphs.

1. Introduction

In recent years, the appearance of multi-agent engineering applications such as Internet-of-Things, Wireless Sensor and Actuator Networks, Smart Energy Grids has stemmed a revival interest in distributed algorithms and in particular distributed convex optimization. Among the very many algorithms available, two of them have attracted the attention of many communities in the past decade, namely the *Alternating Direction Method of Multipliers*, in short ADMM, and the Average Consensus (see the surveys Boyd, Parikh, Chu, Peleato, & Eckstein, 2011 and Garin & Schenato, 2010, for the former and the latter one, respectively). Such interest stands on their wide-range applicability, easy implementation for peer-to-peer architectures, and experimental validations. A large body of literature has appeared regarding the analysis of these two algorithms in terms of rate of convergence, asynchronous implementation and robustness to random communication delays and packet losses. In particular, for what concerns the rate of convergence in a distributed scenario, some notable works include (Iutzeler, Bianchi, Ciblat, & Hachem, 2016; Makhdoumi & Ozdaglar, 2016; Teixeira, Ghadimi, Shames, Sandberg, & Johansson, 2016) for the ADMM and Fagnani and Zampieri (2008) and Xiao and Boyd (2004) for the Average Consensus. These works convey the common belief that ADMM is faster than the Average Consensus but a vis-a-vis comparison between these two algorithms is not provided. A notable exception is given by Erseghe, Zennaro, Dall'Anese, and Vangelista (2011) that compared ADMM against Average Consensus in the specific scenario when the convex problem to be solved is the computation of an arithmetic average. In particular the authors provide expressions for the rate of convergence of the algorithms and showed that ADMM is faster than Average consensus for sparse graphs.

The contribution of this work is twofold. The first is a more compact closed form expression for the rate of convergence of ADMM using a different mathematical machinery than Erseghe et al. (2011), showing that, for highly connected graphs and scalar quadratic cost functions, the Average Consensus algorithm is faster than ADMM.[1] The second contribution is the study of more general multi-variable weighted least-squares problem via numerical simulations which shows that the rate expression obtained for the scalar average, is an optimistic lower bound on the convergence rate of ADMM, since the skewness of the cost function may really compromise the speed of ADMM.

2. Notation and problem formulation

We consider a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of N agents which can communicate among themselves. We only consider *undirected* and

☆ The material in this paper was partially presented at the 15th annual European Control Conference, June 29–July 1, 2016, Aalborg, Denmark. This paper was recommended for publication in revised form by Associate Editor A. Pedro Aguiar under the direction of Editor André L. Tits.

* Corresponding author.
E-mail addresses: bofnicol@dei.unipd.it (N. Bof), carlirug@dei.unipd.it (R. Carli), schenato@dei.unipd.it (L. Schenato).

https://doi.org/10.1016/j.automatica.2018.01.009
0005-1098/

[1] A preliminary version of this result without proof was presented in Bof, Carli, and Schenato (2016).

connected graphs, with all the nodes having a self loop. Given a node $i \in \mathcal{V}$, the set $\mathcal{N}_i$ contains the neighbours of i (i included), that is $\mathcal{N}_i = \{j \,|\, j \in \mathcal{V}\,,\ (i,j) \in \mathcal{E}\}$, and $|\mathcal{N}_i|$ denotes the cardinality of $\mathcal{N}_i$. The adjacency matrix $A_{\mathcal{G}}$ of graph $\mathcal{G}$ is an $N \times N$ matrix with $[A_{\mathcal{G}}]_{ij} = 1$ if $(i,j) \in \mathcal{E}$ and 0 otherwise, where $[A_{\mathcal{G}}]_{ij}$ denotes the element in position (i,j) of matrix $A_{\mathcal{G}}$. A matrix $P \in \mathbb{R}^{N\times N}$ is stochastic if $P\mathbb{1}_N = \mathbb{1}_N$, where $\mathbb{1}_N$ is the all-ones vector of dimension N. A matrix $Q \in \mathbb{R}^{N\times N}$ is said to be *consistent with graph* $\mathcal{G}$ if $[Q]_{ij} > 0 \Leftrightarrow (i,j)$ belongs to $\mathcal{E}$. Since $\mathcal{G}$ is connected and has all the self loops, Q is primitive, and so $[Q^N]_{ij} > 0$ for all i,j. Since $\mathcal{G}$ is an undirected graph, it is possible to find a stochastic matrix P consistent with $\mathcal{G}$ which is also symmetric, and so with eigenvalues $\lambda_1, \ldots, \lambda_N$ which are real and for which it holds $\lambda_1 = 1 > \lambda_2 \geq \cdots \geq \lambda_N > -1$. The *essential spectral radius* (ESR) ρ of such a matrix corresponds to $\max\{|\lambda_2|, |\lambda_N|\} < 1$, and we indicate with $\hat{k}$ the index (equal to 2 or to N) such that $|\lambda_{\hat{k}}| = \rho$.

Each agent is endowed with a quadratic cost function

$$f_i : \mathbb{R}^p \to \mathbb{R}, \qquad f_i(x) = \frac{1}{2}\|x - \theta_i\|^2_{A_i}, \tag{1}$$

where, for $i = 1, \ldots, N$, $A_i \in \mathbb{R}^{p\times p}$ is a positive definite matrix, $\theta_i \in \mathbb{R}^p$, and $\|x\|^2_A = x^\top Ax$. Consider now the global cost function $f : \mathbb{R}^p \to \mathbb{R}$, which is the sum of the cost functions (1) of each agent, $f(x) = \sum_{i=1}^N f_i(x)$. The aim of the agents is to collaborate in order to find the minimizer $x^* \in \mathbb{R}^p$ of $f(x)$ in a distributed way, namely, communicating with their respective neighbours defined by $\mathcal{G}$. This problem corresponds to a distributed weighted least squares which arises in many applications (Carron, Todescato, Carli, & Schenato, 2014; Garin & Schenato, 2010; Xiao & Boyd, 2004). Each agent has therefore to solve the following optimization problem, which has a closed form solution

$$x^* = \operatorname*{argmin}_x f(x) = \left(\frac{1}{N}\sum_{i=1}^N A_i\right)^{-1}\left(\frac{1}{N}\sum_{i=1}^N A_i\theta_i\right). \tag{2}$$

The previous problem can be solved in a distributed fashion via any algorithm able to compute an average, for example *Average Consensus*. An alternative approach is to augment the input domain of (2) and to add additional constraints, and then solve the corresponding problem via Lagrangian-based algorithms such as ADMM.

3. Average Consensus

A popular algorithm to compute an average is the Average Consensus. We introduce this algorithm for the case in which the agents have to evaluate the mean of a set of scalars. The algorithm is obtained by constructing a symmetric stochastic matrix P consistent with the communication graph $\mathcal{G}$. Since P is symmetric and primitive, due to Perron–Frobenius theorem, we have

$$\lim_{t\to\infty} P^t = \frac{1}{N}\mathbb{1}_N\mathbb{1}_N{}^\top. \tag{3}$$

To obtain the mean of the elements of a vector $\mathbf{m} \in \mathbb{R}^N$, it is enough to apply the following iterative scheme

$$\begin{cases}\mathbf{x}(t+1) = P\mathbf{x}(t)\\ \mathbf{x}(0) = \mathbf{m}\end{cases}, \quad t \geq 0.$$

The ith element of vector $\mathbf{x}(t)$, i.e., $x_i(t)$, contains the average of $\mathbf{m}$ estimated at time t by the ith agent.

Introducing the quantities $\bar{m} = \frac{1}{N}\sum_{i=1}^N m_i$ and $\mathbf{x}^* = \bar{m}\mathbb{1}_N$, expression in (3) implies that $\lim_{t\to\infty}\mathbf{x}(t) = \mathbf{x}^*$.

The convergence rate of this algorithm, denoted as ρ_C, is determined by the ESR of matrix P, see Olshevsky and Tsitsiklis (2009). In particular, for a positive constant β depending only on $\mathbf{x}(0)$, it holds

$$\|\mathbf{x}^* - \mathbf{x}(t)\| \leq \beta\rho_C^t, \quad \forall t \geq 0.$$

Here, matrices P are built via the Metropolis–Hastings weights (MHW), Boyd, Diaconis, and Xiao (2004), since they can be calculated locally (each agent only needing the number of its neighbours and their degree). Conversely, building P to have minimal ESR requires the solution of a centralized optimization problem (Boyd et al., 2004). Usually, with MHW, dense graphs far from being bipartite (e.g. random geometric graphs with high distance threshold as in Section 5, or also graphs with many randomly selected edges Boyd et al., 2004) have ρ_C close to 0 (and exactly 0 if $\mathcal{G}$ is complete), while for sparse graphs ρ_C tends to 1. We denote graphs whose corresponding stochastic matrix has a small ESR as well-connected. Problem (2) can be solved by running $p^2 + p$ Average Consensus algorithms, one for each element of A_i and $A_i\theta_i$. Interestingly, the rate of convergence is independent of the A_i's and θ_i's, and it solely depends on the ESR.

4. ADMM

To solve (2) in a distributed way using ADMM, the problem has to be recast defining suitable equality constraints. In particular, with the introduction of the auxiliary vectors $z_i \in \mathbb{R}^p$, $i = 1, \ldots, N$, the problem becomes

$$\begin{cases}\operatorname*{argmin}_{x_i,z_i} \sum_{i=1}^N f_i(x_i)\\ \text{subject to } x_i = z_j, \quad \forall j \in \mathcal{N}_i,\ \forall i \in \{1,\ldots,N\}\,.\end{cases}$$

If $x_1^*, \ldots, x_N^*$ are the solution of the problem above, the constraints assure that $x_i^* = x_j^*$ for all i and j, and so $x_i^* = x^*$ for all i. ADMM exploits the augmented Lagrangian with Lagrangian multipliers $\lambda_{ij} \in \mathbb{R}^p$ and penalty parameters $w_{ij} > 0, i,j = 1, \ldots, N, (i,j) \in \mathcal{E}$:

$$\mathcal{L} = \sum_{i=1}^N \frac{1}{2}\|x_i - \theta_i\|^2_{A_i} + \sum_{i=1}^N\sum_{j\in\mathcal{N}_i}\lambda_{ij}^\top(x_i - z_j) + \frac{1}{2}\sum_{i=1}^N\sum_{j\in\mathcal{N}_i} w_{ij}\|x_i - z_j\|^2.$$

Differently from standard ADMM, we do not impose $w_{ij} = \bar{w}$ for all $i,j = 1, \ldots, N$ (see Erseghe et al., 2011), and we collect all the w_{ij} in matrix $W \in \mathbb{R}^{N\times N}$, with $[W]_{ij} = 0$ if $(i,j) \notin \mathcal{E}$. The standard updates of ADMM (Boyd et al., 2011) are obtained from the Lagrangian imposing first order optimality condition on the x_i's and z_i's and then using an ascent step for variables λ_{ij}'s,

$$x_i(t+1) = \Big(A_i + \sum_{j\in\mathcal{N}_i} w_{ij}I_p\Big)^{-1}\Big(A_i\theta_i + \sum_{j\in\mathcal{N}_i}(w_{ij}z_j(t) - \lambda_{ij}(t))\Big), \tag{4}$$

$$z_j(t+1) = \frac{\sum_{i\in\mathcal{N}_j} w_{ij}x_i(t+1) + \sum_{i\in\mathcal{N}_j}\lambda_{ij}(t)}{\sum_{i\in\mathcal{N}_j} w_{ij}}, \tag{5}$$

$$\lambda_{ij}(t+1) = \lambda_{ij}(t) + w_{ij}\left(x_i(t+1) - z_j(t+1)\right), \tag{6}$$

where I_N is the identity matrix of dimension N. This algorithm converges for any choice of initial conditions for $x_i(0), z_i(0)$ and $\lambda_{ij}(0)$, and for any choice of $w_{ij} > 0$ for all $(i,j) \in \mathcal{E}$. However, its convergence rate strongly depends on the values of the matrices A_i's and the weights w_{ij}. In the following, under a simplified scalar scenario and a particular choice for W, we derive a closed form expression for the best achievable rate of convergence.

4.1. Special case: $A_i = \bar{a}$

We now provide an analytical result for the convergence rate of ADMM for scalar quadratic functions under the assumption $A_i = \bar{a}$, $\bar{a} \in \mathbb{R}$. This corresponds to distributively find the average of N numbers θ_i (collected in vector $\theta \in \mathbb{R}^N$), since the optimizer of