Distributed Augmented Lagrangian Algorithms: Convergence Rate

Dušan Jakovetić[1], José M. F. Moura[2], and João Xavier[1]

[1]Instituto Superior Técnico (IST), Technical University of Lisbon, Lisbon, Portugal
[2]Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA

Abstract— **This paper presents explicit convergence rates for a class of deterministic distributed augmented Lagrangian methods. The expressions for the convergence rates show the dependence on the underlying network parameters. Simulations illustrate the analytical results.**

Index Terms— **distributed optimization, linear convergence rate, augmented Lagrangian, consensus.**

I. Introduction

Recently, there has been much interest in distributed augmented Lagrangian (AL) and alternating direction of multipliers (ADMM) methods. These methods have demonstrated good empirical performance on several distributed signal processing applications, e.g., [1], [2], [3], but, until recently, there has been little understanding of their convergence rates and the rates' dependence on the underlying network.

In this paper, we focus on the distributed optimization problem where each node i in a generic network has a convex, twice continuously differentiable cost $f_i : \mathbb{R}^d \rightarrow \mathbb{R}$, known only to node i. The goal is for each node i to obtain the minimizer $x^\star$ of the sum $\sum_{i=1}^N f_i(x)$ of the nodes' local costs. We consider two different versions of distributed AL algorithms that solve the latter problem, namely distributed AL with nonlinear Jacobi (NJ) primal variable updates, and distributed AL with gradient descent primal variable updates. The former variant is similar to the methods in [4], [1]. (We refer to Section III for the algorithm details.) Our main contribution is to establish for both variants *globally linear convergence rates*, in terms of the number of elapsed per-node communications. Further, we show how the rates depend on the algebraic connectivity of the underlying network. For the AL with NJ updates, the rate $\mathcal{R}$ (the smaller it is, the better) is $1-\Omega\left(\frac{\lambda_2}{\gamma+1}\right)$,[1] where $\lambda_2 = \lambda_2(\mathcal{L})$ is the algebraic connectivity – second smallest eigenvalue of the associated weighted Laplacian matrix $\mathcal{L}$ (the larger the better), and γ is the condition number of the Hessian of the f_i's (the smaller the better). For the AL with gradient-type updates, $\mathcal{R} = 1-\Omega\left(\frac{\lambda_2}{\gamma+1}\frac{\log(1+1/(1+\gamma))}{\log(1+\gamma)+\log(\lambda_2^{-1})}\right)$. Our expressions above explicitly show the joint effect of the "optimization difficulty" (condition number γ) and the degree of the network connectivity (λ_2).

We briefly comment on the literature. Distributed AL and ADMM methods have been recently applied to several signal processing applications, e.g.,[1], [4], [2], [3]. Reference [5] establishes the sublinear $O(1/k)$ rate for a deterministic distributed ADMM method therein, while reference [6] establishes the same rate (in expectation) for an asynchronous distributed ADMM method. These works assume a wider class of the f_i's than the class we assume here but establish much slower rates. Reference [7] establishes a globally linear rate and the dependence on the underlying network for a distributed ADMM and the special case of quadratic f_i's (consensus), while [8] establishes these results for strongly convex f_i's with Lipschitz continuous gradients. With respect to [8], we additionally assume twice continuous differentiability of the f_i's but establish globally linear rates for a *different class* of distributed AL algorithms than [8]. Besides distributed *deterministic* AL methods studied here, in [9] we consider and establish globally linear rates for *randomized* distributed AL methods as well.

The remainder of the paper is organized as follows. The next paragraph introduces notation. Section II introduces the network and optimization models, and Section III presents distributed AL algorithms. Section IV states our convergence rate results for these algorithms. Section V provides simulation examples. Finally, we conclude in Section VI.

We use throughout the following notation. We denote by: $\mathbb{R}^d$ the d-dimensional real coordinate space; a_l the l-th entry of vector a; A_{lm} or $[A]_{lm}$ the entry in the l-th row and m-th column of a matrix A; I, 0, 1, and e_i, respectively, the identity matrix, the zero matrix, the column vector with unit entries, and the i-th column of I; J the $N \times N$ ideal consensus matrix $J := (1/N)1\,1^\top$; $\|\cdot\| = \|\cdot\|_2$ the Euclidean (respectively, spectral) norm of its vector (respectively, matrix) argument; $\lambda_i(\cdot)$ the i-th smallest eigenvalue; $A \succ 0$ means that the Hermitian matrix A is positive definite; $\lfloor a \rfloor$ the integer part of a real scalar a; $\nabla\phi(x)$ and $\nabla^2\phi(x)$ the gradient and Hessian at x of a twice differentiable function $\phi : \mathbb{R}^d \rightarrow \mathbb{R}$, $d \geq 1$. For two positive sequences η_n and χ_n, $\eta_n = O(\chi_n)$ means that $\limsup_{n\rightarrow\infty} \frac{\eta_n}{\chi_n} < \infty$; $\eta_n = \Omega(\chi_n)$ means that $\liminf_{n\rightarrow\infty} \frac{\eta_n}{\chi_n} > 0$; and $\eta_n = \Theta(\chi_n)$ means that $\eta_n = O(\chi_n)$ and $\eta_n = \Omega(\chi_n)$.

The work of the first and third authors is partially supported by: the Carnegie Mellon|Portugal Program under a grant from the Fundação para a Ciência e Tecnologia (FCT), Portugal; by FCT grants CMU-PT/SIA/0026/2009; and by ISR/IST plurianual funding (POSC program, FEDER). The work of the first and second authors is partially supported by NSF under grants CCF-1011903 and CCF-1018509, and by AFOSR grant FA95501010291. D. Jakovetić holds a fellowship from FCT.

[1] See notational conventions in the last paragraph of Section I.

978-1-4799-0248-4/13/$31.00 ©2013 IEEE

II. Problem model

Subsection II-A describes the communication model, while Subsection II-B describes the optimization model.

A. Communication model

We consider a N-node network $\mathcal{G} = (\mathcal{V}, E)$, where $\mathcal{V}$ is the set of nodes, and $E \subset \mathcal{V} \times \mathcal{V}$ is the set of edges.

Assumption 1 The network $\mathcal{G}$ is connected, undirected, and simple (no self/multiple links.)

We denote by O_i the neighborhood set of node i (including i.)

Weight matrix and weighted Laplacian. We associate with graph $\mathcal{G}$ a symmetric, stochastic (rows sum to one and all the entries are non-negative), $N \times N$ weight matrix W, with, for $i \neq j$, $W_{ij} > 0$ if and only if, $\{i, j\} \in E$, and $W_{ii} = 1 - \sum_{j \neq i} W_{ij}$. We require that W is positive definite and that $\lambda_{N-1}(W) < 1$. See [10] how these requirements can be fulfilled beforehand in a distributed way, without knowledge of any global network parameters. Also, denote by $\mathcal{L} := I - W$ the weighted graph Laplacian matrix. The quantity $\lambda_2(\mathcal{L}) \in [0, 1)$ (the larger it is, the better) measures, in a sense, how well connected the network is. For example, for a chain N-node network, $\lambda_2(\mathcal{L}) = \Theta\left(\frac{1}{N^2}\right)$, while, for expander graphs, it stays bounded away from zero as N grows.

B. Optimization model

Nodes solve the unconstrained problem:

$$\text{minimize} \ \sum_{i=1}^{N} f_i(x) =: f(x). \tag{1}$$

The function $f_i : \mathbb{R}^d \to \mathbb{R}$ is known only by node i. We impose the following structure on the f_i's.

Assumption 2 The functions $f_i : \mathbb{R}^d \mapsto \mathbb{R}$ are convex, twice continuously differentiable, and have bounded Hessian, i.e., there exist $0 < h_{\min} \leq h_{\max} < \infty$, such that, for all i:

$$h_{\min} I \preceq \nabla^2 f_i(x) \preceq h_{\max} I, \ \forall x \in \mathbb{R}^d. \tag{2}$$

Under Assumption 2, problem (1) has the unique solution $x^\star$. Denote by $f^\star = \inf_{x \in \mathbb{R}^d} f(x) = f(x^\star)$ the optimal value.

III. Algorithms

In this section, we present our distributed AL algorithms. We first present in detail the algorithm with NJ primal updates; then, we present the algorithm with gradient updates by focusing only on the differences with respect to the former variant.

Distributed AL with NJ primal updates is summarized in Algorithm 1. It has, as tuning parameters, the dual step-size $\alpha > 0$, the AL penalty parameter $\rho > 0$, the number of inner iterations τ, and the weight matrix W (Recall Subsection II-A.) The algorithm operates in two time scales. In the outer iterations k (see (4) in Algorithm 1), each node i updates its dual variable $\eta_i(k) \in \mathbb{R}^d$. In the inner iterations s (see (3) and (4)), each node i updates its primal variable $x_i(k, s)$. At each inner iteration s, each node i broadcasts $x_i(k, s)$ to all its immediate neighbors (see (4)). Outer iterations k do not involve any communications. There are τ inner iterations per each outer iteration k. Note that Algorithm 1 also defines the primal variables $x_i(k)$ at the outer iteration level, as well as certain auxiliary variables $\overline{x}_i(k, s)$ and $\overline{x}_i(k)$. For simplicity, we assume that all nodes use the same initialization of the primal variables: $x_i(0) = x_j(0)$, $\forall i, j$.

Algorithm 1 Distributed AL with NJ updates

```
1: (Initialization) Node $i$ sets $k = 0$, $x_i(k = 0) \in \mathbb{R}^d$, $\overline{x}_i(k = 0) = x_i(0)$, and $\eta_i(k = 0) = 0$.
2: (Inner iterations) Node cooperatively run the nonlinear Jacobi method for $s = 0, 1, ..., \tau - 1$, with $x_i(k, s = 0) := x_i(k)$ and $\overline{x}_i(k, s = 0) := \overline{x}_i(k)$:
       $x_i(k, s+1) = \arg\min_{x_i \in \mathbb{R}^d} \big( f_i(x_i) + (\eta_i(k) - \rho\, \overline{x}_i(k, s))^\top x_i + \frac{\rho \|x_i\|^2}{2} \big)$   (3)
       $\overline{x}_i(k, s+1) = \sum_{j \in O_i} W_{ij}\, x_j(k, s+1)$,   (4)
   and set $x_i(k+1) := x_i(k, s = \tau)$, $\overline{x}_i(k+1) = \overline{x}_i(k, s = \tau)$.
3: (Outer iteration) Node $i$ updates the dual variable $\eta_i(k)$ via:
       $\eta_i(k+1) = \eta_i(k) + \alpha\, (x_i(k+1) - \overline{x}_i(k+1))$.   (5)
4: Set $k \mapsto k + 1$ and go to step 2.
```

Distributed AL with gradient primal updates is the same as the alternative variant, except that the step (3) is replaced with the following:

$$\begin{aligned} x_i(k, s+1) &= (1 - \beta\, \rho)\, x_i(k, s) + \beta\, \rho\, \overline{x}_i(k, s) \quad (6) \\ &- \beta\, (\, \eta_i(k) + \nabla f_i(x_i(k, s))\,)\,, \end{aligned}$$

where $\beta > 0$ is the primal step-size – an additional algorithm parameter.

IV. Linear convergence rates

In this Section, we state and interpret our main results on the convergence rates of the two AL methods. For the proofs, we refer to [9]. Denote by $D_x := \|x_1(0) - x^\star\|$, and $D_\eta := \left(\frac{1}{N} \sum_{i=1}^{N} \|\nabla f_i(x^\star)\|^2\right)^{1/2}$. Also, recall that $\gamma = h_{\max}/h_{\min}$ is the condition number of the f_i's. We have the following Theorem for the AL with NJ updates.

Theorem 1 (AL with NJ updates) Consider Algorithm 1 under Assumptions 1 and 2, and suppose that the algorithm and network parameters satisfy the following:

$$\alpha \leq h_{\min}, \ \left(\frac{\rho}{\rho + h_{\min}}\right)^\tau < \frac{1}{3} \frac{\lambda_2(\mathcal{L})\, h_{\min}}{\rho + h_{\max}}. \tag{7}$$

Then, at any node i, $x_i(k)$ generated by Algorithm 1 converges linearly (in the outer iterations k) to the solution $x^\star$, with rate:

$$\begin{aligned} r_{\mathrm{det,nj}} \ := \ \max\Big\{ & \frac{1}{2} + \frac{3}{2} \left(\frac{\rho}{\rho + h_{\min}}\right)^\tau, \quad (8) \\ & \left(1 - \frac{\alpha\, \lambda_2(\mathcal{L})}{\rho + h_{\max}}\right) + \frac{3\alpha}{h_{\min}} \left(\frac{\rho}{\rho + h_{\min}}\right)^\tau \Big\} < 1, \end{aligned}$$

and there holds:

$$\|x_i(k) - x^\star\| \leq (r_{\mathrm{det,nj}})^k \, \sqrt{N} \, \max\left\{ D_x, \, \frac{2\,D_\eta}{\sqrt{\lambda_2(\mathcal{L})}\,h_{\min}} \right\}.$$

We interpret Theorem 1. First, we can see that the linear convergence rate is guaranteed only for a certain range of the algorithm parameters α, ρ, τ. (See condition (7).) For example, we can take $\rho \leq h_{\min}$, $\alpha = h_{\min}$, and: $\tau \geq \left\lceil \frac{\log\left(\frac{3(1+\gamma)}{\lambda_2(\mathcal{L})}\right)}{\log(2)} \right\rceil$. Second, we can see the interesting dependence of the convergence constant on the quantity D_η. The quantity measures, in a sense, the difficulty of (1) when solved in a distributed way; D_η measures how large are the *local functions' gradients* $\nabla f_i(x^\star)$'s at the *global solution* $x^\star$. If the $\nabla f_i(x^\star)$'s are very small, then the local minimizers $x_i^\star := \arg\min f_i(x)$ are similar to the global minimizer $x^\star$, and hence nodes do not need to cooperate to find a point close to $x^\star$ ("easy" instance of (1).) If, on the other hand, $\nabla f_i(x^\star)$'s are very large, then the $x_i^\star$'s may be very different from $x^\star$ and nodes need cooperation to recover $x^\star$ ("difficult" instance of (1).) Third, although the Theorem states the convergence rate in terms of the outer iterations k, the algorithm converges linearly in the overall number of inner iterations (number of per-node communications), with rate $\mathcal{R} = r_{\mathrm{det,nj}}^{1/\tau}$. Fourth, we can see the explicit dependence of the convergence rate on the algebraic connectivity $\lambda_2(\mathcal{L})$. Specifically, the rate $\mathcal{R} = 1 - \Omega\left(\frac{\lambda_2}{\gamma+1}\right)$ is obtained by setting $\rho = 0$ and $\tau = 1$, which corresponds to the ordinary dual decomposition method.[2]

Theorem 2 (AL with gradient updates) Consider Algorithm 1 where step (3) is replaced with (6), and let Assumptions 1 and 2 hold. Further, suppose that the algorithm and network parameters satisfy the following:

$$\begin{aligned} \alpha &\leq h_{\min}, \, \beta \leq \frac{1}{h_{\max} + \rho} \\ (1 - \beta\,h_{\min})^\tau &< \frac{1}{3}\,\frac{\lambda_2\,h_{\min}}{\rho + h_{\max}}. \end{aligned} \tag{9}$$

Then, at any node i, $x_i(k)$ converges linearly (in the outer iterations k) to the solution $x^\star$, with rate:

$$\begin{aligned} r_{\mathrm{det,grad}} := \; & \max\{\frac{1}{2} + \frac{3}{2}\,(1 - \beta\,h_{\min})^\tau, \qquad (10) \\ & \left(1 - \frac{\alpha\,\lambda_2(\mathcal{L})}{\rho + h_{\max}}\right) + \frac{3\alpha}{h_{\min}}\,(1 - \beta\,h_{\min})^\tau\} < 1, \end{aligned}$$

and there holds:

$$\|x_i(k) - x^\star\| \leq (r_{\mathrm{det,grad}})^k \, \sqrt{N} \, \max\left\{ D_x, \, \frac{2\,D_\eta}{\sqrt{\lambda_2(\mathcal{L})}\,h_{\min}} \right\}.$$

Taking: $\tau = \left\lceil \frac{\log\left(\frac{6(\gamma+1)}{\lambda_2(\mathcal{L})}\right)}{\log\left(\frac{\gamma+1}{\gamma}\right)} \right\rceil$, $\alpha = \rho = h_{\min}$, and $\beta = \frac{1}{\rho + h_{\max}}$, and using Taylor expansions, one obtains the communication rate $\mathcal{R} = r_{\mathrm{det,grad}}^{1/\tau} = 1 - \left(\frac{\lambda_2}{\gamma+1}\,\frac{\log(1+1/(1+\gamma))}{\log(1+\gamma)+\log(\lambda_2^{-1})}\right)$.

V. Simulations

This Section provides a simulation example for the AL and NJ updates and the l_2-regularized logistic losses. Simulations demonstrate the linear convergence rate of the method. We further compare the AL method with NJ updates with the D–NG method in [10]. Simulations suggest that the AL method with NJ updates trades-off communication and computational costs with respect to D–NG, both in the case of larger and smaller condition numbers. (AL with NJ has a lower communication cost and a larger computational cost.) For simulation examples of the AL with gradient updates, we refer to [9].

Simulation setup. The network is geometric: we place nodes uniformly randomly on a unit square and connect the node pairs whose distance is less than a radius. The network has $N = 12$ nodes and 28 links.

Nodes minimize the logistic loss: $\sum_{i=1}^N f_i(x) = \sum_{i=1}^N \left(\log\left(1 + e^{-b_i(a_i^\top x_1 + x_0)}\right) + \frac{\mathcal{P}\,\|x\|^2}{2\,N} \right)$, where $\mathcal{P} > 0$ is the regularization parameter, $x = (x_1^\top, x_0)^\top \in \mathbb{R}^{15}$, $a_i \in \mathbb{R}^{14}$ is the node i's feature vector, and $b_i \in \{-1, +1\}$ is its class label. We take node i's constants $h_{\min,i}$ and $h_{\max,i}$ as: $h_{\min,i} = \frac{\mathcal{P}}{N}$ and $h_{\max,i} = \frac{\mathcal{P}}{N} + \frac{1}{4}\,\|c_i\,c_i^\top\|$. (It can be shown that this choice is in accordance with Assumption 2.) Further, we let $h_{\min} = \min_{i=1,\ldots,N} h_{\min,i}$ and $h_{\max} = \max_{i=1,\ldots,N} h_{\max,i}$. For the problem instance here, the condition number $\gamma = h_{\max}/h_{\min} = 49.55$.

We generate the a_i's independently over i; each entry is drawn from the standard normal distribution. We generate the "true" vector $x^\star = (x_1^{\star\top}, x_0^\star)^\top$ by drawing its entries independently from the standard normal distribution. The class labels are generated as $b_i = \mathrm{sign}\left({x^\star}_1^\top a_i + {x^\star}_0 + \epsilon_i\right)$, where the ϵ_i's are drawn independently from a normal distribution with zero mean and standard deviation 0.001.

The algorithm parameters are as follows. With the AL and NJ updates, we set $\alpha = \rho = h_{\min}$. We set $\tau = 1$ (although our theory does not guarantee linear convergence in such case.) For simulations with theoretical values of τ, we refer to [9]. The weight matrix $W = \frac{1.1}{2} I + \frac{0.9}{2} W_m$, where W_m is the Metropolis weight matrix. (Note that $W \succ 0$.) We initialize the primal and dual variables with zero. With the D–NG method in [10], we set the step-size $\alpha_k = 1/(k+1)$ use the same weight matrix W, and the zero initial estimates. We consider the average relative error in the cost function: $\frac{1}{N}\sum_{i=1}^N \frac{f(x_i) - f^\star}{f(0) - f^\star}$. We compare the methods in terms of: 1) the total number of transmissions (across all nodes), and 2) the total computational time. We use a serial implementation (one processor emulates all nodes.) We count the CPU time across all nodes. At the inner iteration s and outer iteration k of the AL method, we solve (3) via the Nesterov gradient method for strongly convex costs; the implementation details of the method are the same as in [9]. All the Figures are in a semi-log scale.

We consider two scenarios: 1) smaller (better) condition number $\gamma = \frac{h_{\max}}{h_{\min}} = 49.55$; and 2) larger (worse) condition

[2] Although the *upper bound* on the rate in Theorem 1 is the best for $\rho = 0$, the best rate my correspond to the nonzero value of ρ, as demonstrated in [7] for the special case of the quadratic, scalar $f_i(x) = (x - a_i)^2$, $a_i \in \mathbb{R}$.

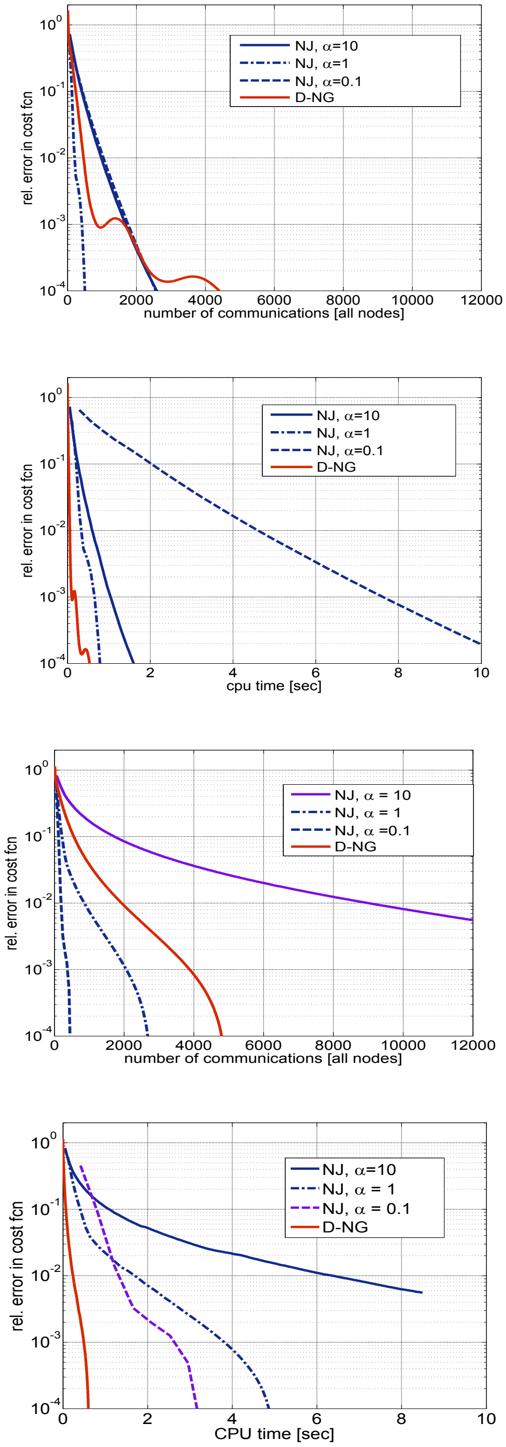

Fig. 1. Average relative error in the cost function $\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)-f^\star}{f(0)-f^\star}$ for the AL with NJ updates method and the D–NG method. The two top figures show the scenario of a smaller condition number $\gamma \approx 49.55$, while the two bottom figures show the scenario of a larger condition number $\gamma \approx 4856$.

number $\gamma \approx 4856$. With the second scenario, we increase the condition number by taking a smaller value of the regularization parameter $\mathcal{P}$. With the AL NJ method, we take $\alpha = \rho \in \{0.01, 0.1, 1, 10\}$, as the optimal choice of α is not known a priori. Figure 1 (first and second from top) are for the smaller condition number, while Figure 1 (third and fourth from top) are for the larger condition number. First, observe that the D–NG method converges sub-linearly in the number of communications, while the AL with NJ updates converges linearly. Second, we can see that, in this implementation example, the D–NG has a lower computational cost, while the AL with NJ has a lower communication cost. Further, we can see that D–NG is not very sensitive to the condition number, neither in terms of communication nor in terms of computational costs. Regarding the AL with NJ, it is not very sensitive in terms of the communication cost, but it is sensitive in terms of the computational cost. The reason is that, for a large (poor) condition number γ, the condition number to solve the local nodes' problems (3) is also poor, and thus the computational cost increases when γ increases.

VI. Conclusion

We considered distributed optimization where N nodes in a generic network minimize the sum $\sum_{i=1}^{N} f_i(x)$ of their individual convex costs. Assuming twice continuously differentiable f_i's with bounded Hessian, we established globally linear convergence rates for two variants of distributed deterministic augmented Lagrangian algorithms. Simulation examples illustrate our results.

References

[1] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links – Part I: Distributed estimation of deterministic signals," *IEEE Trans. Sig. Process.*, vol. 56, no. 1, pp. 350–364, Jan. 2009.

[2] J. Mota, J. Xavier, P. Aguiar, and M. Pueschel, "Distributed basis pursuit," *IEEE Trans. Sig. Process.*, vol. 60, no. 4, pp. 1942–1956, July 2012.

[3] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication," *IEEE Transactions on Signal Processing*, vol. 59, no. 8, pp. 3889–3902, August 2011.

[4] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Transactions on Signal Processing*, vol. 58, no. 11, pp. 5262–5276, November 2010.

[5] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *CDC 2012, IEEE International Conference on Decision and Control*, Maui, Hawaii, Dec. 2012, pp. 5445–5450.

[6] ——, "On the O$(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers," 2013, available at: http://arxiv.org/abs/1307.8254.

[7] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista, "Fast consensus by the alternating direction multipliers method," *IEEE Trans. Sig. Process.*, vol. 59, no. 11, pp. 5523–5537, Nov. 2011.

[8] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization authors," 2013, available at: http://arxiv.org/abs/1307.5561.

[9] D. Jakovetic, J. M. F. Moura, and J. Xavier, "Linear convergence rate of a class of distributed augmented Lagrangain algorithms," 2013, available at: http://arxiv.org/abs/1307.2482.

[10] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *conditionally accepted to IEEE Trans. Autom. Contr.*, Jan. 2013, initial submission, Nov. 2011; available at: http://arxiv.org/abs/1112.2972.