Decentralized Dynamic Optimization Through the Alternating Direction Method of Multipliers

Qing Ling and Alejandro Ribeiro

***Abstract*—This paper develops the application of the alternating directions method of multipliers (ADMM) to optimize a dynamic objective function in a decentralized multiagent system. At each time slot each agent observes a new local objective function and all the agents cooperate to solve the sum objective on the same optimization variable. Specifically, each agent updates its own primal and dual variables and only requires the most recent primal variables from its neighbors. We prove that if each local objective function is strongly convex and has a Lipschitz continuous gradient the primal and the dual variables are close to their optimal values, given that the primal optimal solutions drift slowly enough with time; the closeness is explicitly characterized by the spectral gap of the network, the condition number of the objective function, and the ADMM parameter.**

I. Introduction

We consider a multiagent system composed of n networked agents whose goal at time k is to solve a decentralized dynamic optimization problem with a separable cost of the form

$$\min \sum_{i=1}^{n} f_i^k(\tilde{x}). \tag{1}$$

The variable $\tilde{x} \in \mathbb{R}^p$ is common to all agents that have as their goal the determination of the vector $\tilde{x}^*(k) := \operatorname{argmin} \sum_{i=1}^{n} f_i^k(\tilde{x})$ that solves (1). The problem is decentralized because the cost is separated into convex functions $f_i^k : \mathbb{R}^p \to \mathbb{R}$ known to different agents i and dynamic because the functions f_i^k change over time. The purpose of this paper is to develop the application of the alternating directions method of multipliers (ADMM) to the solution of (1).

Problems having the general structure in (1) arise in a decentralized multiagent system whose task is time-varying. Typical applications include estimating the path of a stochastic process using a wireless sensor network [1] and tracking moving targets and scheduling trajectories in an autonomous team of robots [2], [3]. In the case of static problems, i.e., when the functions $f_i^k(\tilde{x}) = f_i(\tilde{x})$ are the same for all times k, there are many iterative algorithms that enable decentralized solution of (1). Among those we encounter dual subgradient descent algorithms [4] and the ADMM [5], [6], [7], [8]. Both of these algorithms are similar in that they introduce Lagrange multipliers and operate in the dual domain where ascent directions can be computed in a decentralized manner. While their relative performances don't differ dramatically, the ADMM does exhibit better numerical properties in problems with ill-conditioned dual functions [9]. Of particular note, the ADMM has been proved to converge linearly to both, the primal and dual optimal solutions, when all local objective functions are strongly convex and have Lipschitz continuous gradients [10].

Since a dynamic optimization problem can be considered as a sequence of static optimizations any of the methods in [4], [5], [6], [7], [8] can be utilized in their solution. This has indeed been tried in, e.g., [11], [12], where separate time scales are assumed so that the descent iterations are allowed to converge in between different instances of (1). This is not entirely faithful to the time-varying nature of (1) motivating the introduction of algorithms that consider the same time scale for the evolution of the functions f_i^k and the iterations of the distributed optimization algorithm [1], [13], [14], [15].

If the change in the functions f_i^k is sufficiently slow, minor modifications of static algorithms should work reasonably well on keeping track of the time-varying optimal argument $\tilde{x}^*(k)$. In this paper we characterize the norm of the difference between optimal arguments $\tilde{x}^*(k)$ and local estimates of these optimal values for a decentralized implementation of the ADMM. This gap is characterized in terms of the condition number of the functions f_i^k, the spectral gap of the connected network, and the step size of the ADMM algorithm. When the variation vanishes, the ADMM algorithm achieves linear convergence which coincides with the result in [10].

Notation For column vectors $v_1, \ldots, v_n$ use the notation $v := [v_1; \ldots; v_n]$ to represent the stacked column vector v. For a block matrix M use $(M)_{i,j}$ to denote the (i,j)th block. Given matrices $M_1, \ldots, M_n$ use $\operatorname{diag}(M_1, \ldots, M_n)$ to denote the block diagonal matrix whose ith diagonal block is M_i.

II. Problem Formulation and Algorithm Design

Consider a network composed of a set of n agents $\mathcal{V} = \{1, \ldots, n\}$ and a set of m arcs $\mathcal{A} = \{1, \ldots, m\}$, where each arc $e \sim (i,j)$ is associated with an ordered pair (i,j) indicating that i can communicate to j. We assume the networks is connected and that communication is bidirectional so that if $e \sim (i,j)$ there exists another arc $e' \sim (j,i)$. The set of agents adjacent to i is termed its neighborhood and denoted as $\mathcal{N}_i$. The cardinality of this set is the degree d_i of agent i. We define the block arc source matrix $A_s \in \mathbb{R}^{mp \times np}$ where the block $(A_s)_{e,i} = I_p \in \mathbb{R}^{p \times p}$ is an identity matrix if the arc $e \sim (i,j)$ originates at node i and is null otherwise. Likewise, define the block arc destination matrix $A_d \in \mathbb{R}^{mp \times np}$ where

Qing Ling is with Dept. of Automation, University of Science and Technology of China, Hefei, Anhui, China 230026. Alejandro Ribeiro is with Dept. of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, Pennsylvania, USA 19104. Qing Ling is supported by NSFC grant 61004137. Alejandro Ribeiro is supported by NSF CAREER CCF-0952867, NSF CCF-1017454, and AFOSR MURI FA9550-10-1-0567.

978-1-4673-5577-3/13/$31.00 ©2013 IEEE

the block $(A_d)_{e,j} = I_p \in \mathbb{R}^{p\times p}$ if the arc $e \sim (i,j)$ terminates at node j and is null otherwise. Observe that the extended oriented incidence matrix can be written as $E_o = A_s - A_d$ and the unoriented incidence matrix as $E_u = A_s + A_d$. The extended oriented (signed) Laplacian is then given by $L_o = (1/2)E_o^T E_o$, the unoriented (unsigned) Laplacian by $L_u = (1/2)E_u^T E_u$ and the degree matrix containing nodes' degrees d_i in the diagonal is $D = (1/2)(L_o + L_u)$. Denote Γ_L as the largest singular value of L_u and γ_L as the smallest nonzero singular value of L_o; Γ_L and γ_L are both measures of network connectedness.

To solve (1) in a decentralized manner we introduce variables $x_i \in \mathbb{R}^p$ representing local copies of the variable $\tilde{x}$, auxiliary variables $z_{ij} \in \mathbb{R}^p$ associated with each arc $(i,j) \in \mathcal{A}$, and reformulate (1) as

$$\begin{aligned} &\min \ \sum_{i=1}^{n} f_i^k(x_i), \\ &\text{s.t.} \ \ x_i = z_{ij}, \ x_j = z_{ij}, \ \text{for all } (i,j) \in \mathcal{A}. \end{aligned} \tag{2}$$

The constraints $x_i = z_{ij}$ and $x_j = z_{ij}$ imply that for all pairs of agents $(i,j) \in \mathcal{A}$ forming an arc, the feasible set of (2) is such that $x_i = x_j$. For a connected network these local neighborhood constraints further imply that feasible variables must satisfy $x_i = x_j$ for all, not necessarily neighboring, pairs of agents i and j. As a consequence, the optimal local variables in (2) must coincide with the solution of (1), i.e., $x_i^* = \tilde{x}^*$ for all nodes i. We interpret the auxiliary variables z_{ij} as being attached to the arc (i,j) with the purpose of enforcing the equality of the variables x_i and x_j attached to its source agent i and destination agent j.

To simplify discussion define the vector $x = [x_1; \ldots; x_n] \in \mathbb{R}^{np}$ concatenating all variables x_i, the vector $z = [z_1; \ldots; z_m] \in \mathbb{R}^{mp}$ concatenating all variables $z_e = z_{ij}$, and define the aggregate function $f^k : \mathbb{R}^{np} \to \mathbb{R}$ as $f^k(x) := \sum_{i=1}^n f_i^k(x_i)$. Using these definitions and the definitions of the arc source matrix A_s and the arc destination matrix A_d we can rewrite (2) in matrix form as

$$\min \ f^k(x), \qquad \text{s.t.} \ A_s x - z = 0, \ A_d x - z = 0. \tag{3}$$

Further define the matrix $A = [A_s; A_d] \in \mathbb{R}^{2mp\times np}$ stacking the arc source and arc destination matrices A_s and A_d and the matrix $B = [-I_{mp}; -I_{mp}]$ stacking the opposite of two identity matrices so that (3) reduces to

$$\min \ f^k(x), \qquad \text{s.t.} \ Ax + Bz = 0. \tag{4}$$

To introduce the dynamic ADMM for the problem in (2) – and its equivalent forms in (3) and (4) – consider Lagrange multipliers $\alpha_e = \alpha_{ij}$ associated with the constraints $x_i = z_{ij}$ and Lagrange multipliers $\beta_e = \beta_{ij}$ associated with the constraints $x_j = z_{ij}$. Group the multipliers α_e in the vector $\alpha = [\alpha_1; \ldots; \alpha_m] \in \mathbb{R}^{mp}$ and the multipliers β_e in the vector $\beta = [\beta_1; \ldots; \beta_m] \in \mathbb{R}^{mp}$ which are thus associated with the constraints $A_s x - z = 0$ and $A_d x - z = 0$, respectively. Further define $\lambda = [\alpha; \beta] \in \mathbb{R}^{2mp}$ associated with the constraint $Ax + Bz = 0$, a positive constant $c > 0$, and define the augmented Lagrangian function at time k as

$$L_k(x,z,\lambda) = f^k(x) + \lambda^T(Ax + Bz) + \frac{c}{2}\|Ax + Bz\|^2,$$

which differs from the regular Lagrangian by the addition of the quadratic regularization term $(c/2)\|Ax + Bz\|^2$.

The dynamic ADMM proceeds iteratively through alternating minimizations of the Lagrangian $L_k(x,z,\lambda)$ with respect to primal variables x and z followed by an ascent step on the dual variable λ. To be specific, consider arbitrary time k and given past iterates $z(k-1)$ and $\lambda(k-1)$. The primal iterate $x(k)$ is defined as $x(k) := \operatorname{argmin}_x L_k(x, z(k-1), \lambda(k-1))$ and given as the solution of the first order optimality condition

$$\nabla f^k(x(k)) + A^T\lambda(k-1) + cA^T\big[Ax(k) + Bz(k-1)\big] = 0. \tag{5}$$

Using the value of $x(k)$ from (5) along with the previous dual iterate $\lambda(k-1)$ the primal iterate $z(k)$ is defined as $z(k) := \operatorname{argmin}_z L_k(x(k), z, \lambda(k-1))$ and explicitly given by the solution of the first order optimality condition

$$B^T\lambda(k-1) + cB^T\big[Ax(k) + Bz(k)\big] = 0. \tag{6}$$

The dual iterate $\lambda(k-1)$ is then updated by the constraint violation $Ax(k) + Bz(k)$ corresponding to primal iterates $x(k)$ and $z(k)$ in order to compute

$$\lambda(k) = \lambda(k-1) + c\big[Ax(k) + Bz(k)\big]. \tag{7}$$

Observe that the step size c in (7) is the same constant used in (5).

The computations necessary to implement (5)-(7) can be distributed through the network. However, it is also possible to rearrange (5)-(7) so that with proper initialization the updates of the auxiliary variables $z(k)$ are not necessary and the Lagrange multipliers $\alpha \in \mathbb{R}^{mp}$ and $\beta \in \mathbb{R}^{mp}$ can be replaced by a smaller dimension vector $\phi = [\phi_1; \ldots; \phi_n] \in \mathbb{R}^{np}$. We do this in the following proposition before showing that these rearranged updates can be implemented in a decentralized manner. The simplification technique is akin to those used in decentralized implementations of ADMM for static optimization problems; see e.g., [5, Ch. 3], [7].

Proposition 1 *Consider iterates $x(k)$, $z(k)$, and $\lambda(k) = [\alpha(k); \beta(k)]$ generated by recursive application of (5)-(7). Recall the definition of $A = [A_s; A_d]$, the oriented incidence matrix $E_o = A_s - A_d$, the unoriented incidence matrix $E_u = A_s + A_d$, the oriented Laplacian $L_o = (1/2)E_o^T E_o$, the unoriented Laplacian $L_u = (1/2)E_u^T E_u$, and the degree matrix $D = (1/2)(L_o + L_u)$. Require the initial multipliers $\lambda(0) = [\alpha(0); \beta(0)]$ to satisfy $\alpha(0) = -\beta(0)$, the initial auxiliary variables $z(0)$ to be such that $E_o x(0) = 2z(0)$ and further define variables $\phi(k) := E_o^T \alpha(k) \in \mathbb{R}^{np}$. Then, for all times $k > 0$ iterates $x(k)$ can be alternatively generated by the recursion*

$$\begin{aligned} &\nabla f^k(x(k)) + \phi(k-1) + 2cDx(k) - cL_u x(k-1) = 0, \\ &\phi(k) = \phi(k-1) + cL_o x(k). \end{aligned} \tag{8}$$

Proof: See [16]. ■

The iterations in (8) can be implemented in a decentralized manner. To see that this is true consider the component of the update for $x(k)$ corresponding to the variable x_i. Using the definitions of the degree matrix D, the oriented incidence

Algorithm 1 Decentralized Dynamic ADMM at agent i

Require: Initialize local variables to $x_i(0) = 0$, $\phi_i(0) = 0$.
Require: Initialize neighboring variables $x_j(0) = 0$ for all $j \in \mathcal{N}_i$.

1: **for** times k = 1, 2, ... **do**
2: Observe local function f_i^k.
3: Compute local estimate of optimal variable $\tilde{x}^*(k)$ [cf. (9)]

$$\nabla f_i^k(x_i(k)) + 2cd_i x_i(k) = c \sum_{j \in \mathcal{N}_i} \big[x_i(k-1) + x_j(k-1)\big] - \phi_i(k-1).$$

4: Transmit $x_i(k)$ to and receive $x_j(k)$ from neighbors $j \in \mathcal{N}_i$.
5: Update local variable $\phi_i(k)$ as [cf. (10)]

$$\phi_i(k) = \phi_i(k-1) + c \sum_{j \in \mathcal{N}_i} \big[x_i(k) - x_j(k)\big].$$

6: **end for**

matrix E_o, and the unoriented Laplacian L_u we can write this component of the first equality in (8) as

$$\nabla f_i^k(x_i(k)) + 2cd_i x_i(k) = c \sum_{j \in \mathcal{N}_i} \big[x_i(k-1) + x_j(k-1)\big] - \phi_i(k-1). \quad (9)$$

Likewise, using the definitions of the oriented Laplacian L_o the update for $\phi_i(k)$ can be written as

$$\phi_i(k) = \phi_i(k-1) + c \sum_{j \in \mathcal{N}_i} \big[x_i(k) - x_j(k)\big]. \quad (10)$$

At the initialization stage, we choose $\phi(0)$ in the column space of L_o (e.g., $\phi(0) = 0$). This is equivalent to choosing $\lambda(0) = [\alpha(0); \beta(0)]$ such that both $\alpha(0)$ and $\beta(0)$ are in the column space of E_o. Such initialization is necessary for the analysis in Section III.

The decentralized dynamic ADMM algorithm run by agent i is summarized in Algorithm 1. At the initial time $k = 0$ we initialize local variables to $x_i(0) = 0$ and $\phi_i(0) = 0$. Agent i also initializes its local copies of neighboring variables to $x_j(0) = 0$ for all $j \in \mathcal{N}_i$, which is consistent with the initialization at agent j. For all subsequent times agent i goes through successive steps implementing the primal and dual iterations in (9) and (10) as shown in steps 3 and 5 of Algorithm 1, respectively. Implementation of Step 3 requires observation of the local function f_i^k as shown in Step 2 and availability of neighboring variables $x_j(k-1)$ from the previous iteration. Implementation of Step 5 requires availability of current neighboring variables $x_j(k)$, which become available through the exchange implemented in Step 4. This variable exchange also makes variables available for the update in Step 3 corresponding to the following time index.

III. Convergence Analysis

This section analyzes convergence properties of the decentralized dynamic optimization algorithm (9)-(10). We discuss convergence of primal iterates $x(k)$ to the optimal primal variables $x^*(k)$ at time k. We also define the vector $u(k) = [z(k); \alpha(k)]$ which combines primal iterates $z(k)$ and dual iterates $\alpha(k)$ as well as the vector $u^*(k) = [z^*(k); \alpha^*(k)]$ concatenating the unique primal optimal value $z^*(k)$ and a part of an optimal dual variable $\lambda^*(k) = [\alpha^*(k); \beta^*(k)]$ such that $\alpha^*(k)$ lies in the column space of E_o. For distances between $x(k)$ and $x^*(k)$ we consider regular 2-norms and for distances between $u(k)$ and $u^*(k)$ we measure norms with respect to the block diagonal matrix $G = \text{diag}(cI_{mp}, (1/c)I_{mp})$.

Our convergence analysis studies the evolution of the norm $\|u(k) - u^*(k)\|_G$ at subsequent time steps. This result is established in Theorem 1 which relies on the results in lemmas 1 and 2. Lemma 1 is a descent bound on the contraction of the distance $\|u(k-1) - u^*(k-1)\|_G$ between iterate and optimal value at time $k-1$ into the distance $\|u(k) - u^*(k-1)\|_G$ between the optimal value associated with time $k-1$ and the optimal iterate at time k. Lemma 2 bounds the drift of the optimal value $u^*(k-1)$ between subsequent time steps when we are given the value of the drift $\|\tilde{x}^*(k) - \tilde{x}^*(k-1)\|$ between optimal solutions of the original problem in (1). Theorem 1 follows from the triangle inequality and the two bounds in lemmas 1 and 2. Theorem 2 relates the distances $\|u(k-1) - u^*(k-1)\|_G$ and $\|x(k) - x^*(k)\|$ so that the convergence result in Theorem 1 can be translated into a more meaningful statement regarding the suboptimality of primal iterates $x(k)$. The final result in Theorem 3 is stated in the form of a steady state suboptimality gap which is follows from recursive application of the bound in Theorem 2.

Throughout this section we make the following assumptions on the local objective functions f_i^k.

Assumption 1 Local objective functions are differentiable and strongly convex. I.e., for all agents i and times k there exist strictly positive constants $m_{f_i^k} > 0$ such that for all pairs of points x_a and x_b it holds $[\tilde{x}_a - \tilde{x}_b)]^T[\nabla f_i^k(\tilde{x}_a) - \nabla f_i^k(\tilde{x}_b)] \geq m_{f_i^k}\|\tilde{x}_a - \tilde{x}_b\|^2$.

Assumption 2 Local objective functions have Lipschitz continuous gradient. I.e., for all agents i and times k there exist strictly positive constants $M_{f_i^k} > 0$ such that for all pairs of points x_a and x_b it holds $\|\nabla f_i^k(\tilde{x}_a) - \nabla f_i^k(\tilde{x}_b)\|_2 \leq M_{f_i^k}\|\tilde{x}_a - \tilde{x}_b\|_2$.

Assumptions 1 and 2 imply that the sum functions $f^k(x) := \sum_{i=1}^n f_i^k(x_i)$ are also strongly convex with Lipschitz gradients. Indeed, defining $m_f := \min_{i,k} m_{f_i^k} > 0$ as the minimum of all strong convexity constants it follows from Assumption 1 that for all times k and pairs of points x_a and x_b it holds

$$\big[x_a - x_b\big]^T\big[\nabla f^k(x_a) - \nabla f^k(x_b)\big] \geq m_f\|x_a - x_b\|^2 \quad (11)$$

Likewise, defining $M_f := \max_{i,k} M_{f_i^k} > 0$ as the maximum Lipschitz constant it follows from Assumption 2 for all times k and pairs of points x_a and x_b it holds

$$\|\nabla f^k(x_a) - \nabla f^k(x_b)\| \leq M_f\|x_a - x_b\| \quad (12)$$

In the following lemma we utilize (11) and (12) to prove a contraction of the distance $\|u(k-1) - u^*(k-1)\|_G$ between iterate and optimal value at time $k-1$ into the distance $\|u(k) - u^*(k-1)\|_G$ between the optimal value associated with time $k-1$ and the optimal iterate at time k.

Lemma 1 *Consider the dynamic ADMM algorithm defined by* (5)-(7). *At time k, define the vectors* $u(k) = [z(k); \alpha(k)]$ *which* *stacks the current primal and dual variables and* $u^*(k) = [z^*(k); \alpha^*(k)]$ *which stacks the current optimal primal and dual variables and the matrix* $G = \text{diag}(cI_{mp}, (1/c)I_{mp})$. *Let* $\mu > 1$ *be an arbitrary constant and define the corresponding positive number*

$$\delta = \min\left\{\frac{(\mu-1)\gamma_L}{\mu\Gamma_L}, \frac{2cm_f\gamma_L}{c^2\Gamma_L\gamma_L + 4\mu M_f^2}\right\}. \tag{13}$$

Then, the norm with respect to G of the difference between $u(k)$ *and* $u^*(k)$ *decreases by, at least, a factor* $1/\sqrt{1+\delta}$ *of the difference between* $u(k-1)$ *and* $u^*(k)$

$$\|u(k) - u^*(k)\|_G \le \frac{\|u(k-1) - u^*(k)\|_G}{\sqrt{1+\delta}}. \tag{14}$$

Proof: See [16]. ■

The constant δ controls the ratio between $\|u(k) - u^*(k)\|_G$ and $\|u(k-1) - u^*(k)\|_G$. Since $\delta > 0$, $u(k)$ is closer to $u^*(k)$ than $u(k-1)$. Fixing $\|u(k-1) - u^*(k)\|_G$, larger δ means smaller distance from $u(k)$ to $u^*(k)$ and stronger contraction. The same constant δ also appears in analyzing the static ADMM. The impact of the network topology, the objective function, and the step size on δ is discussed in [10].

The following lemma considers how the drift of the optimal primal variables $\tilde{x}^*(k) - \tilde{x}^*(k-1)$ translates into a drift of the optimal vectors $u^*(k-1)$ and $u^*(k)$.

Lemma 2 *Consider the dynamic ADMM algorithm defined by* (5)-(7). *Define the positive constant*

$$\sigma = \sqrt{cm} + \frac{M_f\sqrt{n}}{\sqrt{2c\gamma_L}}, \tag{15}$$

and a time-varying number

$$g(k) = \sigma\|\tilde{x}^*(k) - \tilde{x}^*(k-1)\|, \tag{16}$$

where $\tilde{x}^*(k)$ *and* $\tilde{x}^*(k-1)$ *are the optimal solutions of* (1) *at time* k *and time* $k-1$, *respectively. Define* $u(k)$, $u^*(k)$, *and* G *as in Lemma 1. Then the distance from* $u(k)$ *to* $u^*(k)$ *and the distance from* $u(k-1)$ *to* $u^*(k)$, *both measured by the norm with respect to G, has a gap upper bounded by* $g(k)$

$$\|u(k-1) - u^*(k)\|_G - \|u(k-1) - u^*(k-1)\|_G \le g(k). \tag{17}$$

Proof: See [16]. ■

The gap $g(k)$ describes the drift from $u^*(k-1)$ to $u^*(k)$ on the basis of $u(k-1)$. We expect this gap to be small enough. That is, the difference between the two successive optimal solutions $\tilde{x}^*(k-1)$ and $\tilde{x}^*(k)$ is small enough.

Note that in (16) and (17), $\tilde{x}^*(k-1)$ and $u^*(k-1)$ are undefined when $k = 1$. To address this issue, we can define a virtual initial optimization problem $\min \sum_{i=1}^n f_i^0(\tilde{x})$ such that $\tilde{x}^*(0) = 0$ and $u^*(0) = [z^*(0); \alpha^*(0)] = 0$. Combining Lemma 1 and Lemma 2, we get the following theoretical bound which describes the relationship between $\|u(k) - u^*(k)\|_G$ and $\|u(k-1) - u^*(k-1)\|_G$.

Theorem 1 *Consider the dynamic ADMM algorithm defined by* (5)-(7). *Define* $u(k)$, $u^*(k)$, *and* G *as in Lemma 1, the positive number* δ *as in* (13) *and the time-varying gap* $g(k)$ *as in* (16). *Then the distance between* $u(k)$ *and* $u^*(k)$ *and the distance between* $u(k-1)$ *and* $u^*(k-1)$, *both measured by the norm with respect to G, satisfy*

$$\|u(k) - u^*(k)\|_G \le \frac{\|u(k-1) - u^*(k-1)\|_G}{\sqrt{1+\delta}} + \frac{g(k)}{\sqrt{1+\delta}}. \tag{18}$$

Proof: See [16]. ■

Theorem 1 gives the convergence property of the dynamic ADMM. The convergence is discussed upon u, which is the combination of the auxiliary primal variable z and the dual variable α. Often we are more interested in the convergence with respect to the primal variable x, which is given below.

Theorem 2 *Consider the dynamic ADMM algorithm defined by* (5)-(7). *Define* m_f *as the strong convexity constant of* f^k *in* (11), $u(k)$, $u^*(k)$, *and G as in Lemma 1, and the time-varying gap* $g(k)$ *as in* (16). *The distance between* $x(k)$ *and* $x^*(k)$ *measured by the Euclidean norm and the distance between* $u(k-1)$ *and* $u^*(k-1)$ *measured by the norm with respect to G satisfies*

$$\|x(k) - x^*(k)\| \le \frac{\|u(k-1) - u^*(k-1)\|_G}{\sqrt{m_f}} + \frac{g(k)}{\sqrt{m_f}}. \tag{19}$$

Proof: See [16]. ■

If $\tilde{x}^*(k)$ is a constant for all $k \ge k_0 - 1$, then $g(k) = 0$, for all $k \ge k_0$. In this case Theorem 1 shows that $\{\|u(k) - u^*(k)\|_G\}$ is Q-linearly converging to 0 and Theorem 2 shows that $\{\|x(k) - x^*(k)\|\}$ is R-linearly converging to 0.

On the other hand, if $g(k)$ does not converges to 0 but is smaller than a constant, from Theorem 1 and Theorem 2, we have the following theorem which bounds $\|x(k) - x^*(k)\|$.

Theorem 3 *Consider the dynamic ADMM algorithm defined by* (5)-(7). *Define* m_f *as the strong convexity constant of* f^k *in* (11) *and the corresponding positive numbers* δ *as in* (13). *If the time-varying gap* $g(k)$ *defined in* (16) *is smaller than* g_{max} *for all times* k, *the distance between* $x(k)$ *and* $x^*(k)$ *measured by the Euclidean norm satisfies*

$$\lim_{k\to+\infty} \|x(k) - x^*(k)\| \le \frac{\sqrt{1+\delta}}{\sqrt{m_f}\left[\sqrt{1+\delta} - 1\right]} g_{max}. \tag{20}$$

Proof: See [16]. ■

Recall that $g(k) = \sigma\|\tilde{x}^*(k) - \tilde{x}^*(k-1)\|$ defined in (16) measures the drift between $x^*(k)$ and $x^*(k-1)$. Theorem 3 shows that if this drift is upper bounded by g_{max} over all times k then the difference between $x(k)$ and $x^*(k)$ is also bounded. The constant at the right-hand-side of (20) contains δ and m_f. Larger δ gives tighter bound for $\|x(k) - x^*(k)\|$. The strong convexity constant m_f exists since it controls the drifts of the optimal dual variables.

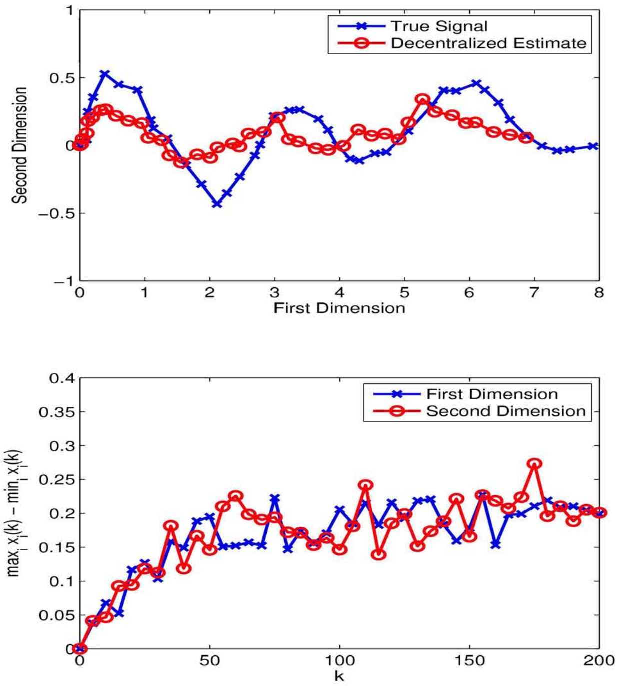

Fig. 1. (TOP) True signal and decentralized estimate of agent 1. (BOTTOM) Maximum distance between decentralized estimates.

IV. Numerical Experiments

This section provides numerical experiments to demonstrate the effectiveness of the proposed dynamic ADMM and validate the theoretical analysis. We consider a connected network with $n = 100$ agents, in which $m = 1810$ arcs (out of all 9900 possible arcs) are randomly chosen to be connected.

At time k agent i measures a true signal $\tilde{x}_0(k)$ through a linear observation function $y_i(k) = H_i(k)\tilde{x}_0(k) + e_i(k)$ where $e_i(k)$ is random noise. We know in advance that the Euclidean distance between $\tilde{x}_0(k)$ and $\tilde{x}_0(k-1)$ is smaller than a threshold ρ. The agents cooperate to recover $\tilde{x}_0(k)$ and the objective function at time k is $f^k(\tilde{x}) = \sum_{i=1}^n \frac{1}{2}\|H_i(k)\tilde{x} - y_i(k)\|^2$ subject to $\|\tilde{x} - \tilde{x}(k-1)\| \leq \rho$. The local objective function of agent i is $f_i^k(\tilde{x}) = \frac{1}{2}\|H_i(k)\tilde{x} - y_i(k)\|^2$ subject to $\|\tilde{x} - x_i(k-1)\| \leq \rho$. This constrained problem is more difficult than the unconstrained one that we have assumed. Accordingly we modify the primal iterate (9) such that agent i can update its $x_i(k)$ from

$$x_i(k) = \arg\min_{x_i} \frac{1}{2}\|H_i(k)x_i - y_i(k)\|^2 + cd_i\|x_i - p_i(k)\|^2, \quad \text{s.t.} \quad \|x_i - x_i(k-1)\| \leq \rho, \tag{21}$$

where $p_i(k) = (1/2d_i)\sum_{j\in\mathcal{N}_i}\left[x_i(k-1) + x_j(k-1)\right] - (1/2cd_i)\phi_i(k-1)$ is a proximal point. In the simulation we let $\tilde{x}(k), e_i(k) \in \mathbb{R}^2$, $H_i(k) \in \mathbb{R}^{2\times 2}$, $\rho = 0.1$, and $c = 1$. We generate elements in $H_i(k)$ following i.i.d. Gaussian distribution $\mathcal{N}(0,1)$ and elements in $e_i(k)$ following i.i.d. Gaussian distribution $\mathcal{N}(0, 0.01)$.

Fig. 1 depicts the simulation results of the dynamic ADMM. The TOP figure compares the true signal, which is close to the centralized solution, and the decentralized estimates of agent 1. The difference between them is bounded throughout the optimization process. This is nontrivial since the optimization problem is dynamic and the inexact solutions $x_i(k-1)$, which are the estimates of $\tilde{x}_0(k-1)$, bring extra uncertainty to the subsequent problems. The BOTTOM figure shows the maximum distance between decentralized estimates of all the agents with respect to the two dimensions. Though each agent optimizes by itself, the agents keep tight consensus. The key is the optimization of the dual variables which guarantees that the consensus constraints don't violate too much.

V. Conclusion

This paper introduces the ADMM to solve a decentralized dynamic optimization algorithm. Traditionally the ADMM is a powerful tool to solve centralized and/or static optimization problems; we show that a minor modification enables it to adapt to the decentralized dynamic cases. We prove that under certain conditions, the differences between the ADMM iterates and the optimal solutions, in both the primal and the dual domains, can be characterized by the drifts between the successive primal optimal solutions.

References

[1] F. Jakubiec and A. Ribeiro, "D-MAP: Distributed maximum a posteriori probability estimation of dynamic systems," IEEE Transactions on Signal Processing, vol. 61, pp. 450–466, 2013
[2] K. Zhou and S. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," IEEE Transactions on Robotics, vol. 27, pp. 678–695, 2010
[3] S. Tu and A. Sayed, "Mobile adaptive networks," IEEE Jounal of Selected Topics in Signal Processing, vol. 5, pp. 649–664, 2011
[4] M. Zargham, A. Ribeiro, A. Jadbabaie, and A. Ozdaglar, "Accelerated dual descent for network optimization," Manuscript
[5] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Second Edition, Athena Scientific, 1997
[6] I. Schizas, A. Ribeiro, and G. Giannakis, "Consensus in ad hoc WSNs with noisy links - Part I: distributed estimation of deterministic signals," IEEE Transactions on Signal Processing, vol. 56, pp. 350–364, 2008
[7] G. Mateos, J. Bazerque, and G. Giannakis, "Distributed sparse linear regression," IEEE Transactions on Signal Processing, vol. 58, pp. 5262–5276, 2010
[8] T. Erseghe, D. Zennaro, E. Dall'Anese, and L. Vangelista, "Fast consensus by the alternating direction multipliers method," IEEE Transactions on Signal Processing, vol. 59, pp. 5523–5537, 2011
[9] 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, vol. 3, pp. 1–122, 2010
[10] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "Linearly convergent decentralized consensus optimization with the alternating direction method of multipliers," Manuscript
[11] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," Proceedings of the IEEE, vol. 95, pp. 215–233, 2007
[12] S. Kar and J. Moura, "Gossip and distributed Kalman filtering: Weak consensus under weak detectability," IEEE Transactions on Signal Processing, vol. 59, pp. 1766–1784, 2011
[13] P. Braca, S. Marano, V. Matta, and P. Willet, "Asymptotic optimalit of running consensus in testing binary hypothesis," IEEE Transactions on Signal Processing, vol. 58, pp. 814–825, 2010
[14] D. Bajovic, D. Jakovetic, J. Xavier, B. Sinopoli, and J. Moura, "Distributed detection via Gaussian running consensus: large deviations asymptotic analysis," IEEE Transatcions on Signal Processing, vol. 59, pp. 4381–4396, 2011
[15] F. Cattivelli and A. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," IEEE Transactions on Automatic Control, vol. 55, pp. 2069–2084, 2010
[16] Q. Ling and A. Ribeiro, "Decentralized dynamic optimization through the alternating direction method of multipliers," Manuscript