BFGS-ADMM for Large-Scale Distributed Optimization

Yichuan Li¹, Yonghai Gong², Nikolaos M. Freris², Petros Voulgaris³ and Dušan Stipanović¹

Abstract—We consider a class of distributed optimization problem where the objective function consists of a sum of strongly convex and smooth functions and a (possibly nonsmooth) convex regularizer. A multi-agent network is assumed, where each agent holds a private cost function and cooperates with its neighbors to compute the optimum of the aggregate objective. We propose a quasi-Newton Alternating Direction Method of Multipliers (ADMM) where the primal update is solved inexactly with approximated curvature information. By introducing an intermediate consensus variable, we achieve a block diagonal Hessian which eliminates the need for inner communication loops within the network when computing the update direction. We establish global linear convergence to the optimal primal-dual solution without the need for backtracking line search, under the assumption that component cost functions are strongly convex with Lipschitz continuous gradients. Numerical simulations on real datasets demonstrate the advantages of the proposed method over state of the art.

I. INTRODUCTION

Distributed optimization acts as the computational engine for a wide range of applications in modern technology, including distributed control [1], power grid management [2], distributed machine learning [3], and resource allocation in sensor networks [4]. We consider a classical problem of minimizing a sum of cost functions and a convex regularizer, aiming at promoting sparsity in the solution structure. In specific:

\[
\begin{align*}
\min_{\hat{x} \in \mathbb{R}^d} \left\{ \sum_{i=1}^{m} f_i(\hat{x}) + g(\hat{x}) \right\},
\end{align*}
\]

where \( f_i : \mathbb{R}^d \to \mathbb{R} \) captures the \( i \)-th agent’s objective of interest and \( g : \mathbb{R}^d \to \mathbb{R} \) is a possibly nonsmooth regularizer, such as the weighted \( \ell_1 \)-norm. In a network of agents, each agent aims to minimize its local cost function \( f_i(\cdot) \) while communicating with its neighbors to cooperatively find the solution of the global problem. Distributed solutions are often pursued to fully utilize the computational power of the agents and reduce the amount of message passing in the network. Efficient communication is deemed especially desirable in scenarios where the number of participants \( m \), and the dimension of the decision variable \( d \) is large, such as in emerging applications of Cyber-Physical systems (CPS) [5]. An archetypal framework of distributed optimization introduces local decision variables at each agent, which are updated locally using private data and exchange variables with neighboring agents. A network-wide solution is achieved by means of asymptotic elimination of the consensus error, i.e., the difference between the values of neighboring agents converges to zero. Such a framework is termed distributed consensus optimization and offers significant flexibility for each agent to select appropriate updating schemes that suit its local hardware environment.

First order methods [6]-[9] constitute popular choices for solving (1) in a distributed fashion. The presence of the nonsmooth regularizer prohibits gradient descent from being implemented directly as \( g(\cdot) \) is not differentiable. Subgradient methods [6] invoke the notion of subdifferential set to compute descent directions while letting agents exchange information over a time-varying topology. Proximal gradient [8] accommodates the nonsmoothness by splitting the objective function and evaluating the proximal mapping associated with the nondifferentiable part. Various acceleration schemes [7], [9] exist for first order methods, that involve storing past gradient and iterate information so as to form a momentum correction term. However, as the size of data grow the condition number of problem (1) tends to get large which causes first order methods to exhibit extremely slow convergence rate. A natural remedy for the aforementioned issues is to consider second order methods [10]-[12]. Through use of function curvature information, second order methods compute descent directions in the objective function level sets that are effective at accelerating the convergence over first order methods. Proximal Newton [12] can be considered as the second order counterpart of the proximal gradient but the associated proximal mapping increases computational burden drastically and renders distributed implementation infeasible as global information is required to compute the Newton step. Moreover, second order methods require solving a linear system to compute the Newton step which induces a computation cost of \( O(d^3) \). Quasi-Newton methods [13] circumvent this procedure by using a finite difference of gradients to approximate the Hessian. The reduced computation costs along with competitive performance have rendered quasi-Newton methods a desirable alternative to second order methods, especially for Large-scale optimization problems.

Primal-dual methods [14]-[15] provide a different perspective for problem (1) within the framework of constrained optimization, where the consensus constraint is explicitly enforced. The Alternating Direction Method of Multipliers (ADMM) [15] falls into this category where the smooth and
nonsmooth parts of the objective are considered separately and iterates are computed sequentially. However, at each iteration of the ADMM, a sub-optimization problem must be solved, which typically induces an expensive computational burden.

**Contributions:** (i) We propose BFGS-ADMM for convex composite optimization where the primal update (typically computationally expensive) is replaced with a quasi-Newton step that does not require solving a linear system of equations. Moreover, by introducing an intermediate consensus variable, we eliminate the need for inner loops within the network when computing the update direction. (ii) We establish global linear convergence rate for the proposed algorithm without backtracking, under the assumption that private cost functions are strongly convex with Lipschitz continuous gradients. (iii) The advantage of BFGS-ADMM over first order methods is analytically established and experimentally demonstrated with numerical simulations on real datasets.

**Notation:** We denote column vectors $x \in \mathbb{R}^d$ with lower case letters and matrices $A \in \mathbb{R}^{n \times m}$ with capital letters. Superscripts are used to denote partitioned vector components and subscripts are used to denote iteration steps, e.g., $x^i_t$ denotes the value of subvector $x_i$ at step $t$. Matrix entries are denoted as $[A]_{ij}$. Unless specified otherwise, $\|x\|$ and $\|A\|$ denote the Euclidean norm of a vector and the corresponding induced norm of a matrix, respectively. We define the norm of a vector associated with a positive definite matrix $P > 0$ as $\|x\|_P := \sqrt{x^\top Px}$, and the set $\{1, \ldots, n\}$ is abbreviated as $[n]$. The proximal mapping associated with the function $g(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}$ is defined as $\prox_{\nu g}(v) = \arg\min_{\theta} \left\{g(\theta) + \frac{\nu}{2}\|\theta - v\|^2\right\}$.

### II. PRELIMINARIES

#### A. Problem formulation

Consider a connected network of $m$ agents captured by a $m$-th order undirected graph: $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, m\}$ is the vertex set, $\mathcal{E}$ contains pairs $(i,j)$ if and only if agent $i$ communicates with agent $j$, and $|\mathcal{E}| = n$ captures the number of communication links in the network. We denote the neighborhood of agent $i$ as $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\}$. We further define the source and the destination matrix (arbitrary ordering suffices since both are given the same value) $A_s, A_d \in \mathbb{R}^{n \times m}$ respectively as follows. Each row of $A_s$ and $A_d$ corresponds to a communication link $\mathcal{E}_k = (i,j)$, for some $k \in [n]$, and $[A_s]_{ki} = [A_d]_{kj} = 1$ with all other entries equal to 0. We reformulate (1) to the consensus framework by introducing local decision variables $x^i$ held by the corresponding agent $i$, along with a set of intermediate consensus variables $\{z^j\}$. Specifically, we define the neighborhood of agent $i$ as $\mathcal{N}_i = \{j : (i,j) \in \mathcal{E}\}$.

\[
\begin{align*}
\minimize_{x^i, \theta, z} & \left\{ \sum_{i=1}^{m} f^i(x^i) + g(\theta) \right\} \\
\text{s.t.} & \quad x^i = z^j, \quad \forall i, j \in \mathcal{N}_i, \\
& \quad x^i = \theta, \quad \text{for some } l \in [m],
\end{align*}
\]  

where the $l$-th agent is chosen arbitrarily to enforce the additional constraint for the nonsmooth regularizer decision variable. The constraints enforce consensus among the network if the underlying graph is connected and, therefore, (2) is equivalent to (1) in the sense that they have the same set of optimal solutions, i.e., $\hat{x}_s = x^1_s = \cdots = x^m_s = \theta = z_s$. Consensus can be achieved by directly enforcing $x^i = x^j$ but having a intermediate consensus variable $z^j$ is crucial in terms of decoupling the primal variable $x^i$ so that computing quasi-Newton steps does not require additional communication within the network. We provide further discussion on this design choice in Section III.}

**Remark 1.** We may compactly express (2) using the aforementioned source and destination matrices as follows:

\[
\begin{align*}
\minimize_{x \in \mathbb{R}^{md}, \theta \in \mathbb{R}^d, z \in \mathbb{R}^m} & \left\{ \mathcal{F}(x) + g(\theta) \right\} \\
\text{s.t.} & \quad Ax = \left[ A_s \otimes I_d \left| A_d \otimes I_d \right| \right] x = \left[ I_n \left| I_n \right. \right] z = Dz, \\
& \quad CX = \theta,
\end{align*}
\]  

where $\otimes$ denotes the Kronecker product and $I$ denotes the identity matrix of the corresponding dimension. We stack local decision variables as $x \in \mathbb{R}^{md} := ([x^1]^\top, \ldots, [x^m]^\top)^\top$ and similarly for the consensus variable $z \in \mathbb{R}^m := ([z]^\top, \ldots, [z]^\top)^\top$, and matrix $A$ and $D$ is obtained by stacking matrices as shown in (3). The matrix $C \in \mathbb{R}^{d \times md} := ([c_1]^\top \otimes I_d$ enforces the last line of constraint of (2) using the coordinate selection vector $c_1 \in \mathbb{R}^m$, whose entries are zero except the $l$-th one being 1. We present some equalities that link our constraint matrices to the topology of the graph in the following.

\[
\begin{align*}
\hat{E}_s &= A_s - A_d, \quad \hat{E}_u = A_s + A_d \tag{4} \\
\hat{E}_s^\top \hat{E}_s &= \hat{L}_s, \quad \hat{E}_u^\top \hat{E}_u = \hat{L}_u \tag{5} \\
\hat{\Delta} &= \frac{1}{2}(\hat{L}_s + \hat{L}_u) = \hat{A}_s^\top \hat{A}_s + \hat{A}_d^\top \hat{A}_d \tag{6}
\end{align*}
\]  

Matrices $\hat{E}_s, \hat{E}_u \in \mathbb{R}^{n \times m}$ stand for the signed and unsigned incidence matrix of the graph, respectively, while $\hat{L}_s, \hat{L}_u \in \mathbb{R}^{n \times m}$ denotes the signed and unsigned Laplacian matrix of the graph, respectively. The diagonal degree matrix of the graph is denoted as $\hat{\Delta}$, whose entries are $\hat{\Delta}^{ii} = |\mathcal{N}_i|$.

**Assumption 1.** Local cost functions $f^i(\cdot)$ are twice differentiable with uniformly bounded Hessian as follows:

\[
m_f I \preceq \nabla^2 f^i(\cdot) \preceq M_f I, \quad \forall i \in [m],
\]  

where $0 < m_f \leq M_f < \infty$. Since $F(x) = \sum_{i=1}^m f^i(x^i)$, $\nabla^2 F(x)$ is block diagonal with the $i$-th block being $\nabla^2 f^i(x^i)$. Therefore, the same bounds apply to the Hessian of $F(x)$ as well, i.e., $m_f I \preceq \nabla^2 F(x) \preceq M_f I$.

#### B. Introduction to BFGS

Quasi-Newton methods [16]-[18] constitute a popular class of methods for accelerating the convergence of optimization methods without directly invoking the Hessian. Quasi-Newton methods seek to approximate the curvature information using consecutive gradient evaluations and iterates differences. More specifically, we define the descent direction $h_t$ as

\[
h_t := -B^{-1}_t \nabla f_t,
\]  

where $B_t$ is a positive definite matrix that approximates the Hessian $\nabla^2 f_t$. Various schemes exist for designing
algorithms that iteratively update $B_t$ including those by Davidson, Fletcher, and Powell (DFP) [16] and Broyden, Fletcher, Goldfarb, and Shannon (BFGS) [17].

In this paper, we focus on the BFGS scheme to select $B_t^{-1}$ as it is observed to work best in the practice both in terms of convergence speed and self-correcting capabilities [18]. The BFGS approximates the Hessian using finite differences of consecutive iterates and gradient evaluations. In specific, define the iterates difference and the gradient difference as follows:

$$s_t = x_{t+1} - x_t, \quad d_t = \nabla f(x_{t+1}) - \nabla f(x_t).$$

Quasi-Newton methods select $B_{t+1}$ so that the secant condition is satisfied, i.e., $B_{t+1}s_t = d_t$, which is motivated by the fact that the real Hessian satisfies this condition as $x_{t+1}$ tends to $x_t$. To ensure $B_{t+1} > 0$, it must hold that $s_t^T d_t > 0$ as can be seen by premultiplying the secant equation with $s_t^T$. For convex objectives, this condition is satisfied automatically.

Note that, however, the secant condition alone is not enough to specify $B_{t+1}$. To uniquely determine $B_{t+1}$, we further require its inverse to be close to the previous value $B_t^{-1}$ in the following sense:

$$\minimize_{B^{-1}} \left\| B^{-1} - B_t^{-1} \right\|_W$$

s.t. $B^{-1} = (B^{-1})^T$, $B^{-1}d_t = s_t$,

where $\left\| A \right\|_W = \left\| W^{1/2}AW^{1/2} \right\|$ stands for the weighted Frobenius norm associated with matrix $W$ whose inverse is the average Hessian [18]. A closed form solution for (9) can be obtained as:

$$B_{t+1}^{-1} = \left( I - \frac{s_t s_t^T}{s_t^T d_t} \right) B_t^{-1} \left( I - \frac{d_t d_t^T}{s_t^T d_t} \right) + \frac{s_t s_t^T}{d_t d_t^T}.$$  \hspace{1cm} (10)

C. Alternating Direction Method of Multipliers

We proceed to introduce the Augmented Lagrangian associated with problem (2) as follows:

$$L(x, z, \theta; y, \lambda) = F(x) + g(\theta) + y^\top (Ax - Dz) + \frac{\lambda^\top (Cx - \theta) + \frac{1}{\mu_1} \left\| Ax - Dz \right\|^2 + \frac{1}{\mu_2} \left\| Cx - \theta \right\|^2, \hspace{1cm} (11)$$

where $y \in \mathbb{R}^{2n}$, $\lambda \in \mathbb{R}^d$ are dual variables corresponding to the two linear constraints in (3). The Alternating Direction Method of Multipliers (ADMM) solves (2) by sequentially minimizing the Augmented Lagrangian over primal/dual variables as:

$$x_{t+1} = \arg\min_x L(x, z_t; \theta_t; y_t), \hspace{1cm} (11a)$$

$$z_{t+1} = \arg\min_z L(x_{t+1}, z, \theta_{t+1}; y_t), \hspace{1cm} (11b)$$

$$\theta_{t+1} = \arg\min_\theta L(x_{t+1}, z_{t+1}; \theta; y_t), \hspace{1cm} (11c)$$

$$y_{t+1} = y_t + \frac{1}{\mu_1} (Ax_{t+1} - Dz_{t+1}) \hspace{1cm} (11d)$$

$$\lambda_{t+1} = \lambda_t + \frac{1}{\mu_2} (Cx_{t+1} - \theta_t) \hspace{1cm} (11e)$$

Note that (11) is a 3-block ADMM which, in general, is not guaranteed to converge for arbitrary values of $\mu_1, \mu_2 > 0$ [19]. We separate the dual variables into $y$ and $\lambda$ for two reasons: (i) Since dual variables accumulate the consensus error within the network as seen in (11d) and (11e), and from that the fact $y$ corresponds to $2n$ constraints while $\lambda$ only involves a single constraint, it is beneficial to separate the two and design different stepsizes $\frac{1}{\mu_1}$ and $\frac{1}{\mu_2}$, which allows for extra freedom for hyperparameters and results in better performance in practice. (ii) In section III, we further show that with appropriate initialization, the storage requirement for $y$ can be reduced by half.

In each iteration of ADMM, a sub-optimization problem has to be solved in (11a) to obtain $x_{t+1}$. This may be computationally expensive and place heavy burden on the system. We therefore propose to perform inexact optimization with the aid of an approximated Hessian using BFGS, aiming to reduce computational costs while maintaining fast convergence speed.

III. ALGORITHMIC DESCRIPTION AND IMPLEMENTATION

We build a local model of the Augmented Lagrangian at step $t$ with respect to the primal variable $x$ as:

$$\hat{L}_t(x) = L_t + \nabla_x L_t^\top (x - x_t) + \frac{1}{2} \left\| x - x_t \right\|_H^2,\hspace{1cm} (12)$$

where $L_t := L(x_t, z_t, \theta_t; y_t, \lambda_t)$. Various algorithms can be designed by choosing different $H_t$. In this paper, we opt to use regularized BFGS approximation of the Augmented Lagrangian Hessian as follows:

$$H_t = B_t + \frac{1}{\mu_1} \Delta + \frac{1}{\mu_2} C^\top C + \epsilon I_{md}, \hspace{1cm} (13)$$

where $\epsilon > 0$ and $\Delta = A^\top A$ is the Kronecker product of the graph degree matrix $\Delta$ and $I_d$. Note that by setting $B_t = \nabla^2 F(x_t)$, we recover the Hessian of the Augmented Lagrangian with respect to $x$, i.e., $H_t = \nabla^2 x L$ in such case. The quadratic form of (12) admits a closed form solution when minimizing over $x$, i.e.,

$$x_{t+1} = x_t - H_t^{-1} \nabla_x L_t.$$  \hspace{1cm} (14)

Step (11b) reduces to solving the following linear system of equations:

$$D^\top y_t + \frac{1}{\mu_1} D^\top (Ax_{t+1} - Dz_{t+1}) = 0. \hspace{1cm} (15)$$

Moreover, by completion of squares, step (11c) is equivalent to evaluating the following proximal mapping associated with $g(\cdot)$ with parameter $\mu_2$:

$$\theta_{t+1} = \prox_{\mu_2 g}(C x_{t+1} + \mu_2 \lambda_t). \hspace{1cm} (16)$$

Dual variables are updated in verbatim as in (11d) and (11e). We proceed to present Lemma 1 which states that with an appropriate initialization, the intermediate variable $\{z_t\}$ evolves on the manifold defined by the column space of $E_u$. Matrices without the hat denotes the Kronecker product of the corresponding matrix with the identity matrix, e.g., $E_u = E_u \otimes I_d$ where $E_u$ is defined in (4).

**Lemma 1.** Recall the signed and unsigned incidence and Laplacian matrices in (4) and (5), respectively. For zero initialization of $(x_t, y_t)$, we can decompose $y_t^\top = [\alpha_t^\top - \alpha_t^\top]$ for all $t \geq 0$. Moreover, the update rules (14), (16), (11d), and (11e) can be simplified as:

$$x_{t+1} = x_t - H_t^{-1} \{ \nabla F(x_t) + E_t^\top \alpha_t + C^\top \lambda_t + \frac{1}{2\mu_1} L_x x_t \} + \frac{1}{\mu_1} C^\top (x_t^\top - \theta_t), \hspace{1cm} (17a)$$

$$\theta_{t+1} = \prox_{\mu_2 g}(x_{t+1} + \mu_2 \lambda_t), \hspace{1cm} (17b)$$

$$\alpha_{t+1} = \alpha_t + \frac{1}{2\mu_1} E_x x_{t+1}, \hspace{1cm} (17c)$$

$$\lambda_{t+1} = \lambda_t + \frac{1}{\mu_2} (x_{t+1}^\top - \theta_{t+1}). \hspace{1cm} (17d)$$
Proof: See Appendix A.

Once $\theta_{t+1}$ is obtained, updates (17c) and (17d) can be performed in parallel and we have effectively transformed a 3-block ADMM in (11) to a 2-block ADMM, which converges under more general settings [20].

Remark 1: The reason for introducing the consensus variable $z$ in (2) is to decouple $x^i$’s so that the Hessian of the Augmented Lagrangian has a block diagonal structure, cf. (13), which is instrumental for computing the quasi-Newton step without additional communication within the network once the gradient of the Augmented Lagrangian is obtained. Note that there is no need to invert any matrix or solve a linear system, and the inverse in (17), is only notational. Instead, we directly approximate each block ($B_i^{-1}$)ii, corresponding to the agent $i$ and form ($H_i^{-1}$)ii as explained in the following. Since $B_i^{-1}$ is block diagonal, we approximate ($B_i^{-1}$)ii using (10) with $d_i = \nabla f^i(x_i^{t+1}) - \nabla f^i(x_i^t)$ and $s_i = x_i^{t+1} - x_i^t$. Moreover, $H_i^{-1} - B_i^{-1}$ is diagonal and constant (time invariant) from (13), which allows for eigendecomposition with components only depending on local information. Specifically, using the Sherman-Morrison formula and defining $C_i^k = (B_i^{-1})^{ii}$, we obtain an iterative formula for ($H_i^{-1}$)ii as follows:

$$C_i^k = C_i^{k-1} - \frac{C_i^k v_i^k (v_i^k)^	op}{1 + (v_i^k)^	op C_i^{k-1} v_i^k},$$

with constant vector $v_i^k$ specified as:

$$v_i^k = \left(\frac{\lambda_i}{\mu_1} + \frac{1}{\mu_2} + \frac{1}{\rho} \right) e_k,$$

where $1_{il} = 1$ if $i = l$ (i.e., if it is the l-th agent that performs the proximal mapping associated with the nonsmooth regularizer $g(.)$) and is zero otherwise, and $e_k \in \mathbb{R}^d$ is a constant vector with all entries equal to 0 except for the k-th one. If consensus is enforced directly as $x^i = x^j$, the resulting Hessian of the Augmented Lagrangian would involve the graph Laplacian matrix as in (11) which couples $x^i$ with its neighbors. In other words, computing quasi-Newton steps would introduce multiple inner communication rounds within the network at each iteration, to approximate the descent direction to a desired accuracy.

We now present a first order variant of BFGS-ADMM, which only uses first order information, so as to demonstrate the merits of using the BFGS approximation, both theoretically and experimentally. Note that the only part of $H_i$ in (13) that depends on the iterate counter is $B_i$, which is an approximation of $\nabla^2 F(x_i)$. Therefore, a first order approximation can be derived by setting $B_i = 0$ in which case $H_i$ in (13) is a constant diagonal matrix. In other words, agent $i$ selects a step size equal to:

$$s_i = \left(\frac{\lambda_i}{\mu_1} + \frac{1}{\mu_2} + \frac{1}{\rho} \right)^{-1}.$$  (20)

Therefore the first order variant updates the primal vector $x_{t+1}$ as:

$$x_{t+1} = x_t - \text{diag}[s_i] \left\{ \nabla F(x_t) + E_s^\top \alpha_t + C^\top \lambda_t + \frac{1}{\rho} L_s x_t + \frac{1}{\mu_2} C^\top (x_i^t - \theta_t) \right\},$$

where $\text{diag}[s_i] \in \mathbb{R}^{md}$ is the diagonal matrix formed by $s_i I_{md}$, and the dual variables ($\theta_{t+1}, \alpha_{t+1}, \lambda_{t+1}$) are updated in the exact same way as in the BFGS-ADMM using (17b). Note that our first order variant encapsulates existing first order methods EXTRA [21] as a special case by setting $s_i = \frac{1}{2}$. Indeed, when $g(\cdot) = 0$, the proximal mapping and the associated dual variables in (17) are not needed and we obtain:

$$x_{t+1} = x_t - \frac{1}{\rho} \left\{ \nabla F(x_t) + E_s^\top \alpha_t + \frac{1}{\mu_2} L_s x_t \right\}.$$  (21)

Taking the difference between the $x_{t+1}$ update and the $x_t$ update defined this way, and substituting the dual update for $\alpha_t$ using (17c), we obtain:

$$x_{t+1} - x_t = 2\left( I - \frac{1}{\rho} L_s \right) x_t - \left( I - \frac{1}{\mu_2} L_s \right) x_{t-1} - \frac{1}{\rho} (\nabla F(x_t) - \nabla F(x_{t-1})).$$  (22)

With appropriate choices of $\mu_1, \mu_2, \rho$, the update defined in (22) is equivalent to EXTRA. We present the comparison between BFGS-ADMM and its first order variant in terms of approximating the exact ADMM in the next section.

We proceed to present the distributed implementation of BFGS-ADMM in Algorithm 1. Note that in the primal update (17a), dual variables are invoked in the form of $E_s^\top \alpha_t$. By defining $\phi_t = E_s^\top \alpha_t \in \mathbb{R}^{md}$ and pre-multiplying both sides of (17c) with $E_s^\top$, we obtain an equivalent algorithm that allows for efficient distributed implementation. We let each agent hold the corresponding pair $(x^i, \phi^i)$ while the l-th agent additionally holds $(\theta, \lambda)$. At each iteration, each agent $i$ begins with calculating $h_i$ using $x_i^t$ from its neighbors as in step 3, where the l-th agent performs an additional update in step 5. Note that step 7 does not require solving a linear system since $(H_i^{-1})^{ii}$ is directly approximated using (18). Primal and dual updates are performed at each agent as in step 8 and step 9, respectively. The l-th agent evaluates an additional proximal mapping in step 11 to update $\theta_{t+1}$, and updates $\lambda_{t+1}$ in step 12. Finally, each agent $i$ updates its $(B_i^{-1})^{ii}$ and $(H_i^{-1})^{ii}$ using (10) and (18), respectively.

Algorithm 1 BFGS-ADMM update

Zero initialization for all variables. Hyperparameters $\mu_1, \mu_2,$ and $\epsilon$. Initialization for $B_0^{-1} = a I_{md}$ for some $a > 0$.

1: for $t = 0, 1, 2, \ldots$ do
2:   for $i \in [m]$ do
3:       Compute $h_i^t = \nabla f^i(x_i^t) + \frac{1}{2 \mu_1} \sum_{j \in \mathcal{N}_i} (x_i^t - x_j^t) + \phi_i^t$.
4:       if $i = l$ then
5:           Update $h_i^{t+1} := \frac{1}{\mu_2} (x_i^{t+1} - \theta_t + \lambda_t)$.
6:       end if
7:       Compute $u_i^t = (H_i^{-1})^{ii} h_i^t$.
8:       Primal update: $x_{i+1}^t = x_i^t - u_i^t$.
9:       Dual update: $\phi_{t+1} = \phi_i^t + \frac{1}{\mu_2} \sum_{j \in \mathcal{N}_i} (x_j^{t+1} - x_i^{t+1})$.
10:  end if
11: end for
12: end for

13: $\theta_{t+1} = \text{prox}_{\mu_2 g}(x_{t+1}^l + \mu_2 \lambda_t)$.
14: $\lambda_{t+1} = \lambda_t + \frac{1}{\mu_2} (x_{t+1}^l - \theta_{t+1})$.
15: Update $(B_{t+1}^{-1})^{ii}$ using (10).
16: Update $(H_{t+1}^{-1})^{ii}$ using (18).
17: end for
18: end for
IV. CONVERGENCE ANALYSIS

Linear convergence rate for ADMM is well established [15]. Since our goal here is to reduce the computational cost of ADMM, the best one can hope for is to reduce the gap between the approximation and the standard ADMM, while maintaining the linear convergence rate. In Lemma 4, we demonstrate the advantage of using BFGS approximation over first order methods by showing that the aforementioned gap decreases faster for BFGS-ADMM. We first state an additional assumption on the nonsmooth regularizer $g(\cdot)$ and the KKT conditions associated with (3) expressed in the primal optimal pair $(x_*, \theta_*)$ and the dual optimal pair $(\alpha_*, \lambda_*)$. Note that we have eliminated $z_*$, as it is shown in Lemma 1 that it is a function of $x_*$. 

Assumption 2: The function $g(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is proper, closed, and convex. Equivalently, $\forall x, y \in \mathbb{R}^d$, the following inequality holds:

$$(x - y)^\top (\partial g(x) - \partial g(y)) \geq 0,$$  \hspace{1cm} (23)

where $\partial g(\cdot)$ denotes the subdifferential set. We proceed to state a non-restrictive assumption that can be easily achieved by regularization.

Assumption 3: There exists positive constant $0 < \nu < \infty$ such that the eigenvalues of $B_t$ are upper bounded, i.e., 

$$B_t < \nu I.$$ 

Remark 2: The upper bound described above can be achieved by setting $B_t^{-1} = B_t^{-1} + \frac{1}{\nu} I$, where $B_t$ comes from the BFGS update formula. Since $B_t^{-1} > 0$ with positive definite initialization, we have $B_t^{-1} > \frac{1}{\nu} I$, i.e., $B_t < \nu I$.

KKT conditions:

$$\nabla F(x_*) + E_s^\top \alpha_* + C^\top \lambda_* = 0 \quad \text{(KKTa)}$$

$$\partial g(\theta_*) - \lambda_* \geq 0 \quad \text{(KKTb)}$$

$$E_s x_0 = 0 \quad \text{(KKTc)}$$

$$x_* - \theta_* = 0 \quad \text{(KKTd)}$$

We proceed to state a technical lemma that describes an inclusion relationship between the subdifferential set of $\partial g(\theta_{t+1})$ and the dual variable $\lambda_{t+1}$.

Lemma 2. Consider the update $\theta_{t+1} = \text{prox}_{\mu_2 g(x_{t+1} + \mu_2 \lambda_*)}$ in (17b). It holds that:

$$\lambda_{t+1} \in \partial g(\theta_{t+1}).$$  \hspace{1cm} (25)

Proof: See Appendix B.

Recall the first order variant of the BFGS-ADMM where the primal variable is updated as in (21) while BFGS-ADMM updates $x_{t+1}$ as in (17a), and both update dual variables as in (17b) and (17c). We proceed to present results which capture their similarities and differences in the following.

Lemma 3. If Assumptions 1-2 hold, then the iterates generated by BFGS-ADMM and its first order variant both satisfy the following equation:

$$e_t = \nabla F(x_t) + B_t (x_{t+1} - x_t) - \nabla F(x_{t+1}).$$  \hspace{1cm} (27)

for its first order variant,

$$e_t = \nabla F(x_t) - \nabla F(x_{t+1}).$$  \hspace{1cm} (28)

Proof: See Appendix C.

By comparing (27) and (28), we see that the difference between using BFGS vs. first order approximation lies in how $\nabla F(x_{t+1})$ is approximated at step $t$. Moreover, if the sub-optimization problem is solved exactly as in (11), by optimality condition, equation (25) holds with $e_t = 0$.

Lemma 4. Consider the bound for $\nabla^2 F(x)$ in (7) and the $e_t$ introduced in Lemma 3. If Assumption 3 holds, an upper bound can be established as follows.

For BFGS-ADMM,

$$\|e_t\| \leq \|B_{t+1} - B_t\| \|x_{t+1} - x_t\|$$  \hspace{1cm} (29)

for its first order variant as in (21),

$$\|e_t\| \leq M_f \|x_{t+1} - x_t\|.$$  \hspace{1cm} (30)

Proof: See Appendix D.

Remark 3: Lemma 4 is a standalone result whose proof does not require the convergence of the algorithm. In Theorem 1, we establish that BFGS-ADMM converges linearly which means $\lim_{t \to \infty} \|B_{t+1} - B_t\| = 0$, since both $d_t, s_t$ approach 0 as $t \to \infty$ in (10). Therefore, $\|e_t\| = O(\|x_{t+1} - x_t\|)$ for the BFGS-ADMM while $\|e_t\| = O(\|x_{t+1} - x_t\|)$ for its first order variant. The variable $e_t$ captures the approximation error vs. exact solution of the primal problem ($e_t = 0$ for this case), whence Lemma 4 unravels that the proposed has superior tracking of the exact ADMM, which is the key attribute to yield superior linear convergence rate over its first-order variant.

Lemma 5. Consider the update in (17c) and (17d). With zero initialization, the stacked vector $[\alpha_0^T \lambda_0^T]^\top \in \mathbb{R}((n+1)d)$ lies in the column space of $X^\top = [E_s^\top C^\top]^\top \in \mathbb{R}((n+1)d)\times nd$. Moreover, there exists a unique optimal $[\alpha_0^T \lambda_0^T]^\top$ in the column space of $X^\top$ and, denoting $\sigma_{\text{min}}^+$ as the smallest positive eigenvalue of $X^\top X$, the following inequality holds:

$$\sigma_{\text{min}}^+ \left(\|\alpha_{t+1} - \alpha_*\|^2 + \|\lambda_{t+1} - \lambda_*\|^2\right) \leq \left\|E_s^\top (\alpha_{t+1} - \alpha_*) + C^\top (\lambda_{t+1} - \lambda_*)\right\|^2.$$  \hspace{1cm} (31)

Proof: See Appendix E.

We introduce the following scaling matrix defined in terms of the hyperparameters $\epsilon, \mu_1, \mu_2$ and the graph topology (as captured by $L_u$), $H = \frac{1}{\epsilon \mu_1} L_u + \epsilon I = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{\mu_1} & 0 & 0 \\ 0 & 2 \mu_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$

Theorem 1. Consider the update in (17). Define $u_t^\top = [x_t^\top, \theta_t^\top, \alpha_t^\top, \lambda_t^\top]$ and $u_*$ the corresponding optimum. Denote the smallest positive eigenvalue of $E_s^\top C^\top$ as $\sigma_{\text{min}}^+$, the smallest and largest eigenvalue of $L_u$ as $\sigma_{\text{min}}^+$ and $\sigma_{\text{max}}^+$, respectively. Denoting $\kappa = \frac{M_f}{m_f}$, for arbitrary constants $\beta, \gamma, \phi, \rho > 1$, and $\zeta \in \{\frac{m_f + M_f}{2m_f M_f}, \frac{M_f}{2m_f M_f}\}$, $\epsilon > 2(m_f + M_f)\kappa$, the iterates converge linearly as follows:

$$\|u_{t+1} - u_*\|_H^2 \leq \frac{1}{1 + 2\zeta}\|u_t - u_*\|_H^2.$$  \hspace{1cm} (32)
where
\[
\delta = \min \left\{ \frac{2m_f M_f}{m_f + M_f}, \frac{\rho}{2} \left( \mu_2 (\sigma_{\min}^2 + \rho_1 \varrho_1) + \rho_1 \varrho_1 \right), \frac{\rho_1 \varrho_1}{\rho} \right\},
\]

and
\[
\left( \frac{3\beta-1}{2\beta} \right) (\frac{3\rho_1 \varrho_1}{\rho}) \left( \frac{\sigma_{\min}^2 + \rho_1 \varrho_1}{\rho} \right),
\]

and
\[
\frac{2\rho_1 \varrho_1}{\rho_1 \varrho_1},
\]

Moreover, using Lemma 3, we rewrite the RHS (right-hand side) of (34) as
\[
\text{RHS} = \left( x_{t+1} - x_\star \right)^T \frac{1}{2m_f} (L_u + e(I)) \left( x_{t+1} - x_t \right)
\]

Using Lemma 2, we have (\latex\text{RHS}) \leq \left( \theta_{t+1} - \theta_\star \right)^T (\lambda_{t+1} - \lambda_\star) \geq 0, \text{ where the inequality follows from Lemma 2 and Assumption 1. Similarly, (37)},

\[
\text{RHS} = \left( x_{t+1} - x_\star \right)^T \frac{1}{2m_f} (L_u + e(I)) \left( x_{t+1} - x_t \right)
\]

Substituting (36) and (37) into (35), we obtain
\[
\text{RHS} = \left( x_{t+1} - x_\star \right)^T \left( \lambda_{t+1} - \lambda_\star \right)
\]

Moreover, using (17d) and (KKTd), we obtain
\[
\left( x_{t+1} - x_\star \right)^T \left( \mu_1 \left( \alpha_{t+1} - \alpha_\star \right) \right)
\]

Using the fact that (35) - (37), we have
\[
\left( \mu_1 \left( \alpha_{t+1} - \alpha_\star \right) \right)^T (\lambda_{t+1} - \lambda_\star)
\]

By setting \( q_t = B_t^2 s_t \), we obtain
\[
\left( q_t \sigma_{\min}^{-1} \right)^2 \leq \left( \lambda_{t+1} - \lambda_\star \right)^2
\]

Denoting \( \mu = \max \{ \mu_1, \mu_2 \} \), then combining (43) - (45), we have:

\[
\left( \lambda_{t+1} - \lambda_\star \right)^2 \leq \frac{2\mu_1 \left( \alpha_{t+1} - \alpha_\star \right)^2 + \mu_2 \left( \lambda_{t+1} - \lambda_\star \right)^2 + \frac{1}{\rho_1} \left( \theta_{t+1} - \theta_\star \right)^2}{\rho_1 \varrho_1}
\]

Furthermore, from (46) and (47), it suffices to show that the following inequality holds for some \( \delta > 0 \):

\[
\delta \left| u_{t+1} - u_\star \right|^2 \leq \left( \frac{2m_f M_f}{m_f + M_f} \right) \left| x_{t+1} - x_\star \right|^2 + \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \left| \nabla F(x_{t+1}) - \nabla F(x_\star) \right|^2
\]

Using (46) and (47), it suffices to show that the following inequality holds for some \( \delta > 0 \):

\[
\delta \left| u_{t+1} - u_\star \right|^2 \leq \left( \frac{2m_f M_f}{m_f + M_f} \right) \left| x_{t+1} - x_\star \right|^2 + \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \left| \nabla F(x_{t+1}) - \nabla F(x_\star) \right|^2
\]

where \( \delta = \min \left\{ \frac{2m_f M_f}{m_f + M_f}, \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \right\}, \text{ and (48)}, \) it suffices to show that \( \delta \left| u_{t+1} - u_\star \right|^2 \leq \left( \frac{2m_f M_f}{m_f + M_f} \right) \left| x_{t+1} - x_\star \right|^2 + \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \left| \nabla F(x_{t+1}) - \nabla F(x_\star) \right|^2
\]

Moreover, for any \( \zeta > 0 \), the inequality holds:

\[
\delta \left| u_{t+1} - u_\star \right|^2 \leq \left( \frac{2m_f M_f}{m_f + M_f} \right) \left| x_{t+1} - x_\star \right|^2 + \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \left| \nabla F(x_{t+1}) - \nabla F(x_\star) \right|^2
\]

Finally, using (49) and (50), we obtain:

\[
\delta \left| u_{t+1} - u_\star \right|^2 \leq \left( \frac{2m_f M_f}{m_f + M_f} \right) \left| x_{t+1} - x_\star \right|^2 + \frac{\rho_1 \varrho_1}{\rho_1 \varrho_1} \left| \nabla F(x_{t+1}) - \nabla F(x_\star) \right|^2
\]

which is satisfied if \( \delta \) is chosen as in (51).
nonskin dataset and the ijcnn1

where the matrix of PG-EXTRA is generated using the Laplacian based edge is drawn independently from a Bernoulli(\(\gamma\)) distribution; \(\gamma\) is the number of data points held at each agent and \((w_j, y_j) \in \mathbb{R}^d \times \{0, 1\}\) are training samples of dimension \(d\) and binary labels, respectively. We consider two real datasets from LIBSVM\(^{(a)}\): the skin_noskin dataset and the ijcnn1 dataset. We take 5,000 data points from each dataset with dimension \(d = 3\) and \(d = 22\), respectively. A random binomial graph with \(m\) agents is drawn (each edge is drawn independently from a Bernoulli(\(p\)) distribution; \(p = 0.2\) was used in all cases). The mixing matrix for P2D2 is generated using the Metropolis rule while the mixing matrix of PG-EXTRA is generated using the Laplacian based constant edge weight matrix. In all cases, the \(l_1\)-norm weight \(\gamma = 2 \times 10^{-6}\). We plot the average relative cost error defined as 

\[
\frac{1}{m} \sum_{i=1}^{m} F(x^i_t) - F(x^i_0)
\]

for different network sizes. As it can be seen in Figure 1 and 2, the BFGS-ADMM method demonstrates significant speed up in both datasets compared to other methods. Note that our first order variant also shows advantages compared to other first order methods, due to the fact that in our first order approximation, step size selection also takes into account the number of neighbors each agent has as in \(^{(20)}\).

V. EXPERIMENTS

In this section we present some numerical experiments of the proposed method compared to distributed first order methods: P2D2 \(^{(22)}\) and PG-EXTRA \(^{(7)}\), for solving the following distributed logistic regression problem:

\[
\minimize_{x \in \mathbb{R}^d} F(x) = \left\{ \frac{1}{m} \sum_{i=1}^{m} f^i(x) + \gamma ||x||_1 \right\},
\]

where \(f^i(x) = \frac{1}{m_i} \sum_{j=1}^{m_i} \ln \left( 1 + e^{w_j^T x} \right) + (1 - y_j)w_j^T x \), and \(m_i\) is the number of data points held at each agent and \((w_j, y_j) \in \mathbb{R}^d \times \{0, 1\}\) are training samples of dimension \(d\) and binary labels, respectively. We consider two real datasets from LIBSVM\(^{(1)}\): the skin_noskin dataset and the ijcnn1 dataset. We take 5,000 data points from each dataset with dimension \(d = 3\) and \(d = 22\), respectively. A random binomial graph with \(m\) agents is drawn (each edge is drawn independently from a Bernoulli(\(p\)) distribution; \(p = 0.2\) was used in all cases). The mixing matrix for P2D2 is generated using the Metropolis rule while the mixing matrix of PG-EXTRA is generated using the Laplacian based constant edge weight matrix. In all cases, the \(l_1\)-norm weight \(\gamma = 2 \times 10^{-6}\). We plot the average relative cost error defined as 

\[
\frac{1}{m} \sum_{i=1}^{m} F(x^i_t) - F(x^i_0)
\]

for different network sizes. As it can be seen in Figure 1 and 2, the BFGS-ADMM method demonstrates significant speed up in both datasets compared to other methods. Note that our first order variant also shows advantages compared to other first order methods, due to the fact that in our first order approximation, step size selection also takes into account the number of neighbors each agent has as in \(^{(20)}\).

REFERENCES

[1] Y. Li, D. Stipanović, P. Voulgaris, and Z. Gu, “Decentralized model predictive control of urban drainage systems,” WSEAS Transactions on Systems and Control, vol. 14, pp. 247–256, 2019.

[1\(^{(1)}\)] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[2] T. Huang, N. M. Freris, P. R. Kumar, and L. Xie, “A synchrophasor data-driven method for forced oscillation localization under resonance conditions,” IEEE Transactions on Power Systems, vol. 35, no. 5, pp. 3927–3939, 2020.

[3] R. Bekkerman, M. Bilenko, and J. Langford, “Scaling up machine learning: Parallel and distributed approaches,” in Proceedings of the 17th ACM SIGKDD International Conference Tutorials, 2011.

[4] N. M. Freris, H. Kowshik, and P. R. Kumar, “Fundamentals of large sensor networks: Connectivity, capacity, clocks, and computation,” Proceedings of the IEEE, vol. 98, no. 11, pp. 1828–1846, 2010.

[5] K. Kim and P. R. Kumar, “Cyber–physical systems: A perspective at the centennial,” Proceedings of the IEEE, vol. 100, pp. 1287–1308, 2012.

[6] A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multi-agent optimization,” IEEE Transactions on Automatic Control, vol. 54, no. 1, pp. 48–61, 2009.

[7] W. Shi, Q. Ling, G. Wu, and W. Yin, “A proximal gradient algorithm for decentralized composite optimization,” IEEE Transactions on Signal Processing, vol. 63, no. 22, pp. 6013–6023, 2015.

[8] N. Parikh and S. Boyd, “Proximal algorithms,” Foundations and Trends in Optimization, vol. 1, no. 3, pp. 123–239, 2014.

[9] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, “DLM: Decentralized linearized alternating direction method of multipliers,” IEEE Transactions on Signal Processing, vol. 63, no. 15, pp. 4051–4064, 2015.

[10] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, “A decentralized second-order method with exact linear convergence rate for consensus optimization,” IEEE Transactions on Signal and Information Processing over Networks, vol. 2, no. 4, pp. 507–522, 2016.

[11] Y. Li, N. M. Freris, P. Voulgaris, and D. Stipanović, “D-SOP: Distributed second order proximal method for convex composite optimization,” in Proceedings of the 2020 American Control Conference, 2020, pp. 2844–2849.

[12] J. D. Lee, Y. Sun, and M. A. Saunders, “Proximal Newton-type methods for minimizing composite functions,” SIAM Journal on Optimization, vol. 24, no. 3, pp. 1420–1443, 2014.

[13] J. E. Dennis, Jr. and J. J. Moré, “Quasi-newton methods, motivation and theory,” SIAM Review, vol. 19, no. 1, pp. 46–89, 1977.

[14] P. Latafat, N. M. Freris, and P. Patrinos, “A new randomized block-coordinate primal-dual proximal algorithm for distributed optimization,” IEEE Transactions on Automatic Control, vol. 64, no. 10, pp. 4050–4065, 2019.

[15] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, “On the linear convergence of the ADMM in decentralized consensus optimization,” IEEE Transactions on Signal Processing, vol. 62, no. 7, pp. 1750–1761, 2014.

[16] W. C. Davidon, “Variable metric method for minimization,” SIAM Journal on Optimization, vol. 1, no. 1, pp. 1–17, 1991.

[17] D. Goldfarb, “A family of variable-metric methods derived by variational means,” Mathematics of Computation, pp. 23–26, 1970.

[18] J. Nocedal and S. J. Wright, Numerical Optimization, 2nd ed. Springer, 2006.

[19] C. Chen, B. He, Y. Ye, and X. Yuan, “The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent,” vol. 155, no. 1–2, 2016.

[20] T. Lin, S. Ma, and S. Zhang, “Global convergence of unmodified 3-block ADMM for a class of convex minimization problems,” Journal of Scientific Computing, vol. 76, no. 1, pp. 69–88, 2018.

[21] W. Shi, Q. Ling, G. Wu, and W. Yin, “EXTRA: An exact first-order algorithm for decentralized consensus optimization,” SIAM Journal on Optimization, vol. 25, no. 2, pp. 944–966, 2015.

[22] S. Alghunaim, K. Yuan, and A. H. Sayed, “A linearly convergent proximal gradient algorithm for decentralized optimization,” in Proceedings of the 2020 American Control Conference, 2020, pp. 2844–2849.

APPENDIX

Proof of Lemma 1

From the construction of \(C = (c^l)^T \otimes I_d \in \mathbb{R}^{d \times md}\), where \(c^l\) selects the \(l\)-th coordinate, we stress that the update \(^{(16)}\) only involves the subvector \(x^l_{t+1}\) as it can be seen by explicitly writing \(^{(16)}\) as:

\[\lambda_{t+1} = \lambda_t + \frac{1}{\rho^l}(x^l_{t+1} - \theta_{t+1}).\]
We decompose the dual variable as $y^T = [\alpha^T \beta^T]$, where $\alpha, \beta \in \mathbb{R}^{nd}$. Recalling $D^T = [I_{nd} I_{nd}]$ and premultiplying the dual update (11d) with $D^T$, we obtain

$$D^Ty_{t+1} = D^Ty_t + \frac{1}{\mu_1}D(Ax_t - Dz_{t+1}).$$

Using (15), we conclude that $D^Ty_{t+1} = 0$ for $t \geq 0$, i.e., $\beta_{t+1} = -\alpha_{t+1}$. With zero initialization, we can therefore express $y_t$ as $y_t = \begin{bmatrix} \alpha_t \\ -\beta_t \end{bmatrix}$, for all $t \geq 0$. This shows that it is redundant to maintain both $\alpha$ and $\beta$ since they sum to 0 for all $t \geq 0$. Therefore, we can rewrite (11d) as:

$$\alpha_{t+1} = \alpha_t + \frac{1}{\mu_1} (Ax_t - z_{t+1}), \quad (48)$$

$$-\beta_{t+1} = -\beta_t + \frac{1}{\mu_1} (Ax_t - z_{t+1}). \quad (49)$$

After taking the difference of the above two equations and dividing by 2, we obtain:

$$\alpha_{t+1} = \alpha_t + \frac{1}{\mu_1} E_u x_t, \quad (50)$$

where we have used the fact that $E_u = A_x - A_d$. Similarly, by summing (48) and (49), we obtain $z_{t+1} = \frac{1}{2} E_u x_t$. With zero initialization for $x_t$, it is redundant to keep $z_t$ since we can compute it as:

$$z_t = \frac{1}{2} E_u x_t \forall t \geq 0. \quad (51)$$

Recalling the approximated Hessian in (13), we express the primal update (14) as

$$x_{t+1} = x_t - H^{-1}_t \{\nabla F(x_t) + A^T y_t + C^T \lambda_t + \frac{1}{\mu_1} A^T (Ax_t - Dz_t) + \frac{1}{\mu_2} C^T (C x_t - \theta_t)\}. \quad (52)$$

Since $E^s_x = A^s_x - A^s_d$, $E^u_x = A^s_u + A^s_d$, and $L_u = E^u_u E^u_x$ as in (14) and (51), along with the decomposition of $y_t$ and the equality in (51), we have

$$A^T y_t = E^s_y \alpha_t, \quad (53)$$

$$\frac{1}{\mu_1} A^T D z_t = \frac{1}{2\mu_1} L_u x_t. \quad (54)$$

Using (4), we can rewrite

$$\frac{1}{\mu_1} A^T Ax_t - \frac{1}{2\mu_1} L_u x_t = \frac{1}{2\mu_1} (2\Delta - L_u) x_t = \frac{1}{2\mu_1} L_u x_t. \quad (55)$$

Substituting (53)-(55) into (52), we obtain the primal updates as in (17a).

**Proof of Lemma 2**

The update (17b) is equivalent to

$$\theta_{t+1} = \argmin_{\theta} \left\{ g(\theta) + \frac{1}{2\mu_2} \|x^T_{t+1} + \mu_2 \lambda_t - \theta\|^2 \right\}. \quad (56)$$

From the optimality condition, it holds that

$$0 \in g(\theta_{t+1}) - \frac{1}{\mu_2} (x^T_{t+1} + \mu_2 \lambda_t - \theta_{t+1}). \quad (57)$$

Substituting the dual update $\lambda_{t+1} = \lambda_t + \frac{1}{\mu_2} (x^T_{t+1} - \theta_{t+1})$ into (56), we obtain

$$0 \in \partial g(\theta_{t+1}) - \lambda_{t+1}. \quad (58)$$

After re-arranging, we obtain the desired.

**Proof of Lemma 3**

Note that the difference between BFGS-ADMM and its first order variant lies in $H_t$ (13). For the first order variant, $H_t = \frac{1}{\mu_1} \Delta + \frac{1}{\mu_2} C^T C + e I_{md}$, i.e., $B_t$ is identically zero. By rearranging (17a), we obtain the following equation:

$$\nabla F(x_t) + E^s_x \alpha_t + C^T \lambda_t + \frac{1}{2\mu_1} L_u x_t + \frac{1}{\mu_2} C^T (C x_t - \theta_t) + (B_t + \frac{1}{\mu_1} \Delta + \frac{1}{\mu_2} C^T C + e I_{md})(x^T_{t+1} - x_t) = 0. \quad (57)$$

Using the dual update (17c), we have

$$E^s_x \alpha_t + \frac{1}{2\mu_1} L_u x_t = E^s_x \alpha_{t+1} - \frac{1}{2\mu_1} L_u (x^T_{t+1} - x_t). \quad (58)$$

Since $2\Delta = L_u + e I_{md}$, it holds that

$$\frac{1}{\mu_1} (x^T_{t+1} - x_t) = \frac{1}{2\mu_1} L_u (x^T_{t+1} - x_t). \quad (59)$$

Substituting (58) and (59) into (57), we obtain

$$\nabla F(x_t) + B_t(x^T_{t+1} - x_t) + \frac{1}{2\mu_1} (L_u + e I_{md})(x^T_{t+1} - x_t) + C^T \left\{ \lambda_t + \frac{1}{\mu_2} (x^T_{t+1} - \theta_t) \right\} + E^s_x \alpha_{t+1} = 0. \quad (60)$$

Using the dual update (17d), we have

$$\lambda_{t+1} = \lambda_t + \frac{1}{\mu_2} (x^T_{t+1} - \theta_t). \quad (61)$$

Substituting (61) into (60) and using the definition of $e_t = \nabla F(x_t) + B_t(x^T_{t+1} - x_t) - \nabla F(x_{t+1})$, we obtain

$$\nabla F(x_{t+1}) + \frac{1}{\mu_1} (L_u + e I)(x^T_{t+1} - x_t) + C^T \{ \lambda_{t+1} + \frac{1}{\mu_2} (\theta_{t+1} - \theta_t) \} + E^s_x \alpha_{t+1} + e_t = 0. \quad (62)$$

After subtracting (KKTa) from (62), we obtain the desired where $B_t = 0$ for the first order variant.

**Proof of Lemma 4**

For BFGS-ADMM, since $B_{t+1}$ satisfies the secant equation:

$$B_{t+1}(x^T_{t+1} - x_t) = \nabla F(x^T_{t+1}) - \nabla F(x_t),$$

and (29) follows by applying the Cauchy-Schwarz inequality. For the first order variant, the Lipschitz continuity of $\nabla F(x)$ implies that $\forall x, y \in \mathbb{R}^{nd}, \|\nabla F(x) - \nabla F(y)\| \leq M_f \|x - y\|$. The claim in (30) follows.

**Proof of Lemma 5**

Consider the update (17c) and (17d). It can be written compactly as:

$$\begin{bmatrix} \alpha_{t+1} \\ \lambda_{t+1} \end{bmatrix} = \begin{bmatrix} \alpha_t \\ \lambda_t \end{bmatrix} + \frac{1}{\mu_2} \begin{bmatrix} E^s_x \quad 0 \\ C \end{bmatrix} \begin{bmatrix} x^T_{t+1} - \theta_{t+1} \end{bmatrix}, \quad (57)$$

where $0$ is a zero vector of dimension $nd$. Therefore, it suffices to prove the vector $[0^T \theta^T_{t+1}]^T$ lies in the column space of $X^T$. Note that since the graph is connected, $E^s_x r = 0$ if $r \in \mathbb{R}^{md}$ is in consensus, i.e., $r^1 = r^2 = \cdots = r^m \in \mathbb{R}^d$. By letting $r^i = \theta_{t+1}$ for all $i$, we have: $E^s_x r^T = 0$, which shows that $[0^T \theta^T_{t+1}]^T$ lies in the column space of $X^T$. To simplify the notation we denote $[\alpha^T, \lambda^T]^T = w_t$. To prove the existence of optimal $w_t$ in the column space of $X^T$, consider the (KKTa) satisfied with any optimal $w_t$,

$$\nabla F(x_t) + w_t = 0. \quad (57)$$

The projection of $w_t$ into the column space of $X^T$, denoted as $w_*$, also satisfies (KKTa) since their differences lie in the kernel of $X$, i.e., $X(w_t - w_*) = 0$. The uniqueness of $w_*$ can be established by contradiction. Let $w_1 = X^T r_1$ and $w_2 = X^T r_2$ be two optimal stacked dual vector that lie in the column space of $X^T$, $r_1 \neq r_2$. Since $F(\cdot)$ is strongly convex, from (KKTa), it holds that $\nabla F(x_t) + X^T r_1 = 0$, $\nabla F(x_t) + X^T r_2 = 0$. After taking the difference, we have $XX^T (r_1 - r_2) = 0$. But $XX^T = E^s_x + C^T C = L_u + C^T C > 0$. Therefore $r_1 - r_2 = 0$, contradiction. Inequality (31) follows from the fact that $w_{t+1} - w_*$ is orthogonal to the kernel of $X$. 
