PARALLELIZABLE ALGORITHMS FOR THE SELECTION OF GROUPED VARIABLES*

Gonzalo Mateos, Juan Andrés Bazerque and Georgios B. Giannakis

Dept. of ECE, Univ. of Minnesota, 200 Union Street SE, Minneapolis, MN 55455, USA

ABSTRACT

Well-appreciated in statistics for its ability to select relevant *grouped* features (factors) in linear regression models, the group-Lasso estimator has been fruitfully applied to diverse signal processing problems including RF spectrum cartography and robust layered sensing. These applications motivate the *distributed* group-Lasso algorithm developed in this paper, that can be run by a network of wireless sensors, or, by multiple processors to balance the load of a single computational unit. After reformulating the group-Lasso cost into a separable form, it is iteratively minimized using the method of multipliers to obtain parallel per agent and per factor estimate updates given by vector soft-thresholding operations. Through affordable inter-agent communication of sparse messages, the local estimates provably *consent* to the global group-Lasso solution. Specializing to a single agent network, or, to univariate factors, efficient (distributed) Lasso solvers are rediscovered as a byproduct.

Index Terms— Sparsity, linear regression, (group-)Lasso, parallel optimization, distributed estimation.

1. INTRODUCTION

Consider the classical problem of linear regression, where a vector $\mathbf{y} \in \mathbb{R}^n$ of observations is given along with a matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ of inputs. Suppose the p features are split into N_f disjoint *factors* (groups of features) such that the coefficient vector is $\boldsymbol{\beta} = [\boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_{N_f}']' \in \mathbb{R}^p$, where $'$ denotes transposition and $\boldsymbol{\beta}_f$ corresponds to the coefficients of factor f. The *group* least-absolute shrinkage and selection operator (Lasso) [1] is a model selection and estimation technique used to select relevant factors in linear regression, and yields

$$\hat{\boldsymbol{\beta}}_{\text{glasso}} := \arg\min_{\boldsymbol{\beta}} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \mu \sum_{f=1}^{N_f} \|\boldsymbol{\beta}_f\|_2 \quad (1)$$

where $\mu \geq 0$ is a tuning parameter typically chosen via model selection techniques such as cross-validation (CV); see e.g., [1, 2]. If $\mu = 0$, no sparsity is enforced since (1) reduces to LS. As μ increases, more sub-vector estimates $\boldsymbol{\beta}_f$ become zero due to the effect of the group sparsity-encouraging penalty, and the corresponding factors drop out of the model. When $N_f = p$, (1) becomes the Lasso [3] that performs variable – rather than factor – selection.

Finding $\hat{\boldsymbol{\beta}}_{\text{glasso}}$ requires solving (iteratively) for any given value of μ a second-order cone program (SOCP). While standard SOCP solvers can be invoked to this end, an increasing amount of effort has been put recently into developing fast algorithms that capitalize on the unique properties of the group-Lasso; see e.g. [1], [4], [5], [6], [7].

Typically, the training set is assumed to be centrally available, so that it can be jointly processed to obtain $\hat{\boldsymbol{\beta}}_{\text{glasso}}$. However, collecting all data in a central location may be prohibitive in timely applications of interest. In-network-based (group-)Lasso estimators find application in e.g., robust layered sensing [8], and in the sensing task of cognitive radio networks [9, 10]. In other cases such as the Internet or collaborative inter-laboratory studies, agents providing private data for the purpose of e.g., fitting a sparse model, may not be willing to share their training data but only the learning results. Distributed subgradient methods are applicable to sparse linear regression [11] as well, but are typically slow.

Having this context in mind, the present paper develops a consensus-based distributed algorithm for the group-Lasso, which can be specialized for the Lasso as well. Problem (1) is recast as a convex *constrained* minimization in Section 2, and is iteratively optimized using the alternating-direction method of multipliers (AD-MoM) [12, p. 253]. This way, provably convergent parallel recursions are derived to update each agent's local estimate, that entail simple vector soft-thresholding operations (Section 3). This is possible by capitalizing on the closed form solution that (1) admits in the orthonormal case [7], [4], and evidences the factor-level sparsity encouraging property of group-Lasso. On a per iteration basis, agents only exchange their current local estimate with their neighbors. By specializing to a dummy single agent network, a novel centralized group-Lasso solver is obtained in Section 4 as a byproduct. Different from [1] and [4], the algorithm here can handle non orthonormal matrix $\mathbf{X}$, and does not require an inner Newton-Raphson recursion per iteration. By comparing the centralized algorithm with its distributed counterparts of Section 3, it is shown that the latter effectively split the computational burden across agents.

*Work in this paper was supported by the NSF grants CCF-0830480 and ECCS-0824007.

2. PROBLEM STATEMENT AND PRELIMINARIES

Consider J networked agents that are capable of performing some local computations, as well as exchanging messages among neighbors. An agent should be understood as an abstract entity, possibly representing a sensor node in a WSN, a router monitoring Internet traffic, a hospital, insurance company or laboratory involved in e.g., a medical study; or a sensing radio from a next-generation mobile communications technology. The network is naturally modeled as an undirected graph $\mathcal{G}(\mathcal{J},\mathcal{E})$, where the vertex set $\mathcal{J}:=\{1,\ldots,J\}$ corresponds to the agents, and the edges in $\mathcal{E}$ represent pairs of agents that can communicate. Agent $j\in\mathcal{J}$ communicates with its single-hop neighboring agents in $\mathcal{N}_j$, and the size of the neighborhood is denoted by $|\mathcal{N}_j|$. The graph $\mathcal{G}$ is assumed connected, i.e., there exists a (possibly multihop) path that joins any pair of agents in the network.

For the purpose of estimating an unknown vector $\boldsymbol{\beta}=[\boldsymbol{\beta}_1',\ldots,\boldsymbol{\beta}_{N_f}']'\in\mathbb{R}^p$, each agent $j\in\mathcal{J}$ has available a local vector of observations $\mathbf{y}_j\in\mathbb{R}^{n_j}$ as well as its own matrix of inputs $\mathbf{X}_j\in\mathbb{R}^{n_j\times p}$. Agents collaborate to form the wanted group-Lasso estimator (1) in a distributed fashion, which can be rewritten as

$$\hat{\boldsymbol{\beta}}_{\text{glasso}}:=\arg\min_{\boldsymbol{\beta}}\frac{1}{2}\sum_{j=1}^{J}\|\mathbf{y}_j-\mathbf{X}_j\boldsymbol{\beta}\|_2^2+\mu\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_f\|_2 \quad (2)$$

with $\mathbf{y}:=[\mathbf{y}_1',\ldots,\mathbf{y}_J']'\in\mathbb{R}^n$ with $n:=\sum_{j=1}^{J}n_j$, and $\mathbf{X}:=[\mathbf{X}_1',\ldots,\mathbf{X}_J']'\in\mathbb{R}^{n\times p}$. In lieu of a central controller, the goal of this paper is to develop a distributed solver of (2) based on in-network processing of the local training sets $\{\mathbf{y}_j,\mathbf{X}_j\}_{j\in\mathcal{J}}$. On a per iteration basis the algorithm should comprise: (i) a communication step where agents exchange messages with their neighbors; and (ii) a simple update step where each agent uses this information to refine its local estimate. An additional desirable property is that the collection of local estimates should eventually *consent* to the global solution $\hat{\boldsymbol{\beta}}_{\text{glasso}}$, namely, the estimate that would be obtained if the entire training data set were centrally available.

2.1. Consensus-based reformulation of group-Lasso

To distribute the cost in (2), replace the *global* variable $\boldsymbol{\beta}$ which couples the per-agent summands, with *local* variables $\{\boldsymbol{\beta}_j\in\mathbb{R}^p\}_{j=1}^J$ representing candidate estimates of $\boldsymbol{\beta}$ per agent. It is now possible to reformulate (2) as a convex *constrained* minimization problem

$$\left\{\hat{\boldsymbol{\beta}}_j\right\}_{j=1}^{J}:=\arg\min_{\{\boldsymbol{\beta}_j\}}\frac{1}{2}\sum_{j=1}^{J}\left[\|\mathbf{y}_j-\mathbf{X}_j\boldsymbol{\beta}_j\|_2^2+\frac{2\mu}{J}\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_{jf}\|_2\right]$$
$$\text{s. t.}\qquad \boldsymbol{\beta}_j=\boldsymbol{\beta}_{j'},\ j\in\mathcal{J},\ j'\in\mathcal{N}_j \quad (3)$$

where $\boldsymbol{\beta}_j=\left[\boldsymbol{\beta}_{j1}',\ldots,\boldsymbol{\beta}_{jN_f}'\right]'$, $j\in\mathcal{J}$. The equality constraints directly effect local agreement between neighboring CRs. Since the communication graph $\mathcal{G}$ is assumed connected, these constraints also ensure *global* consensus a fortiori, meaning that $\boldsymbol{\beta}_j=\boldsymbol{\beta}_{j'}$, $\forall j,j'\in\mathcal{J}$. As a direct consequence of this observation, it follows that problems (2) and (3) are equivalent, i.e., $\hat{\boldsymbol{\beta}}_{\text{glasso}}=\hat{\boldsymbol{\beta}}_j$, $\forall\, j\in\mathcal{J}$.

Problem (3) will be modified further for the purpose of reducing the computational complexity of the resulting algorithm. To this end, for a given $\mathbf{a}\in\mathbb{R}^p$ consider the problem

$$\min_{\boldsymbol{\beta}}\left[\frac{1}{2}||\boldsymbol{\beta}||_2^2-\mathbf{a}'\boldsymbol{\beta}+\mu\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_f\|_2\right],\quad \boldsymbol{\beta}:=[\boldsymbol{\beta}_1',\ldots,\boldsymbol{\beta}_{N_f}']' \quad (4)$$

and notice that it is separable in the N_f subproblems

$$\min_{\boldsymbol{\beta}_f}\left[\frac{1}{2}||\boldsymbol{\beta}_f||_2^2-\mathbf{a}_f'\boldsymbol{\beta}_f+\mu\|\boldsymbol{\beta}_f\|_2\right],\quad \mathbf{a}:=[\mathbf{a}_1',\ldots,\mathbf{a}_{N_f}']'. \quad (5)$$

Interestingly, each of these subproblems admits a closed-form solution as given in the following lemma.

Lemma 1: *The minimizer $\boldsymbol{\beta}_f^\star$ of* (5) *is obtained via the vector soft-thresholding operator $\mathcal{T}_\mu(\cdot)$ defined by*

$$\boldsymbol{\beta}_f^\star=\mathcal{T}_\mu(\mathbf{a}_f):=(\|\mathbf{a}_f\|_2-\mu)_+\frac{\mathbf{a}_f}{\|\mathbf{a}_f\|_2} \quad (6)$$

where $(\cdot)_+:=\max\{\cdot,0\}$.

Problem (4) is an instance of group-Lasso (1) when $\mathbf{X}'\mathbf{X}=\mathbf{I}_p$, and $\mathbf{a}:=\mathbf{X}'\mathbf{y}$. As such, result (6) can be viewed as a particular case of the operators in [4] and [7]. However it is worth to prove Lemma 1 directly, since in this case the special form of (5) renders the proof neat in its simplicity.

Proof: It will be argued that the solver of (5) takes the form $\boldsymbol{\beta}_f^\star=t\mathbf{a}_f$ for some scalar $t\geq 0$. This is because among all $\boldsymbol{\beta}_f$ with the same ℓ_2-norm, the Cauchy-Schwarz inequality implies that the maximizer of $\mathbf{a}_f'\boldsymbol{\beta}_f$ is colinear with (and in the same direction of) $\mathbf{a}_f$. Substituting $\boldsymbol{\beta}_f=t\mathbf{a}_f$ into (5) renders the problem scalar in t, with solution $t^\star=(\|\mathbf{a}_f\|-\mu)_+/(\|\mathbf{a}_f\|)$ completing the proof. □

In order to take advantage of the result in Lemma 1, auxiliary variables $\boldsymbol{\gamma}_j$, $j\in\mathcal{J}$ are introduced as copies of $\boldsymbol{\beta}_j$. Upon introducing appropriate constraints $\boldsymbol{\gamma}_j=\boldsymbol{\beta}_j$ that guarantee the equivalence of the formulations, problem (3) can be recast as

$$\min_{\{\boldsymbol{\beta}_j,\boldsymbol{\gamma}_j,\boldsymbol{\gamma}_j^{j'}\}}\frac{1}{2}\sum_{j=1}^{J}\left[\|\mathbf{y}_j-\mathbf{X}_j\boldsymbol{\gamma}_j\|_2^2+\frac{2\mu}{J}\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_{jf}\|_2\right] \quad (7)$$
$$\text{s. t.}\qquad \boldsymbol{\beta}_j=\boldsymbol{\gamma}_j^{j'}=\boldsymbol{\beta}_{j'},\ j\in\mathcal{J},\ j'\in\mathcal{N}_j$$
$$\boldsymbol{\gamma}_j=\boldsymbol{\beta}_j,\ j\in\mathcal{J}.$$

The additional set of dummy variables $\{\boldsymbol{\gamma}_j^{j'}\}$ is inserted for technical reasons that will become apparent in the ensuing section, and will be eventually eliminated.

3. DISTRIBUTED GROUP-LASSO ALGORITHM

The distributed group-Lasso algorithm is constructed by optimizing (7) using the alternating direction method of multipliers (AD-MoM) [12]. In this direction, associate Lagrange multipliers $\mathbf{v}_j, \bar{\mathbf{v}}_j^{j'}$ and $\check{\mathbf{v}}_j^{j'}$ with the constraints $\boldsymbol{\gamma}_j = \boldsymbol{\beta}_j$, $\boldsymbol{\beta}_j = \boldsymbol{\gamma}_j^{j'}$ and $\boldsymbol{\beta}_{j'} = \boldsymbol{\gamma}_j^{j'}$ respectively, and consider the augmented Lagrangian with parameter $c > 0$

$$\begin{aligned}\mathcal{L}_c\left[\{\boldsymbol{\beta}_r\}, \boldsymbol{\gamma}, \boldsymbol{v}\right] =& \frac{1}{2}\sum_{j=1}^{J}\left[\|\mathbf{y}_j - \mathbf{X}_j\boldsymbol{\gamma}_r\|_2^2 + \frac{2\mu}{J}\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_{jf}\|_2\right] \\ &+ \sum_{j=1}^{J}\left[\mathbf{v}_j'(\boldsymbol{\beta}_j - \boldsymbol{\gamma}_j) + \frac{c}{2}\|\boldsymbol{\beta}_j - \boldsymbol{\gamma}_j\|_2^2\right] \\ &+ \sum_{j=1}^{J}\sum_{j'\in\mathcal{N}_j}\left[(\check{\mathbf{v}}_j^{j'})'(\boldsymbol{\beta}_j - \boldsymbol{\gamma}_j^{j'}) + \frac{c}{2}\|\boldsymbol{\beta}_j - \boldsymbol{\gamma}_j^{j'}\|_2^2\right] \\ &+ \sum_{j=1}^{J}\sum_{j'\in\mathcal{N}_j}\left[(\bar{\mathbf{v}}_j^{j'})'(\boldsymbol{\beta}_{j'} - \boldsymbol{\gamma}_j^{j'}) + \frac{c}{2}\|\boldsymbol{\beta}_{j'} - \boldsymbol{\gamma}_j^{j'}\|_2^2\right] \quad (8)\end{aligned}$$

where variables are grouped as $\boldsymbol{\gamma} := \{\boldsymbol{\gamma}_j, \{\boldsymbol{\gamma}_j^{j'}\}_{j'\in\mathcal{N}_j}\}_{j\in\mathcal{J}}$ and multipliers $\boldsymbol{v} := \{\mathbf{v}_j, \{\check{\mathbf{v}}_j^{j'}, \bar{\mathbf{v}}_j^{j'}\}_{j'\in\mathcal{N}_j}\}_{j\in\mathcal{J}}$.

Application of the AD-MoM to the problem at hand consists of a cycle of $\mathcal{L}_c$ minimizations in block-coordinate descent fashion w.r.t. $\{\boldsymbol{\beta}_j\}$ firstly, and $\boldsymbol{\gamma}$ secondly, together with an update of the multipliers per iteration $k = 0, 1, 2, \ldots$. Omitting the details that can be found in [10, Appendix D], the four main properties of this procedure that are instrumental to the resulting algorithm can be highlighted as:

[P1] Thanks to the introduction of the local copies $\boldsymbol{\beta}_j$ and the dummy variables $\boldsymbol{\gamma}_j^{j'}$, the minimizations of $\mathcal{L}_c$ w.r.t. both $\{\boldsymbol{\beta}_j\}$ and $\boldsymbol{\gamma}$ decouple per agent j, thus enabling distribution of the algorithm. Moreover, the constraints in (7) involve variables of neighboring agents only, which allows the required communications to be local within each agent's neighborhood.

[P2] Introduction of the variables $\boldsymbol{\gamma}_j$ separates the quadratic cost $\|\mathbf{y}_j - \mathbf{X}_j\boldsymbol{\gamma}_j\|_2^2$ from the group-Lasso penalty $\sum_{f=1}^{N_f}\|\boldsymbol{\beta}_{jf}\|_2$. As a result, minimization of (8) w.r.t. $\boldsymbol{\beta}_j$ takes the form of (4), which admits a closed-form solution via the vector soft-thresholding operator $\mathcal{T}_\mu(\cdot)$ in (6).

[P3] Minimization of (8) w.r.t. $\boldsymbol{\gamma}$ consists of an unconstrained quadratic problem, which can also be solved in closed form. In particular, the optimal $\boldsymbol{\gamma}_j^{j'}$ at iteration k takes the value $\boldsymbol{\gamma}_j^{j'}(k) = (\boldsymbol{\beta}_j(k) + \boldsymbol{\beta}_{j'}(k))/2$, and thus can be eliminated.

[P4] It turns out that it is not necessary to carry out updates of the Lagrange multipliers $\{\bar{\mathbf{v}}_j^{j'}, \check{\mathbf{v}}_j^{j'}\}_{j'\in\mathcal{N}_j}$ separately, but only of their sums which are henceforth denoted by $\mathbf{p}_j := \sum_{j'\in\mathcal{N}_j}(\bar{\mathbf{v}}_j^{j'} + \check{\mathbf{v}}_j^{j'})$. Hence, there is one price $\mathbf{p}_j$ per agent $j = 1, \ldots, J$, which can be updated locally.

Algorithm 1 : DGLasso

Initialize to zero $\{\boldsymbol{\beta}_j(0), \boldsymbol{\gamma}_j(0), \mathbf{p}_j(-1), \mathbf{v}_j(-1)\}_{j\in\mathcal{J}}$, and locally run:
for $k = 0, 1, \ldots$ **do**
  Transmit $\boldsymbol{\beta}_j(k)$ to neighbors in $\mathcal{N}_j$.
  Update $\mathbf{p}_j(k)$ using (9).
  Update $\mathbf{v}_j(k)$ using (10).
  Update $\boldsymbol{\beta}_j(k+1)$ using (11).
  Update $\boldsymbol{\gamma}_j(k+1)$ using (12).
end for

Building on these four features, it is established in [10, Appendix D] that the proposed AD-MoM scheme boils down to four parallel recursions run locally per agent, where $f = 1, \ldots, N_f$ in (11) and $\mathbf{M}_j := c\mathbf{I}_p + \mathbf{X}_j'\mathbf{X}_j$ in (12)

$$\mathbf{p}_j(k) = \mathbf{p}_j(k-1) + c\sum_{j'\in\mathcal{N}_j}[\boldsymbol{\beta}_j(k) - \boldsymbol{\beta}_{j'}(k)] \quad (9)$$

$$\mathbf{v}_j(k) = \mathbf{v}_j(k-1) + c[\boldsymbol{\beta}_j(k) - \boldsymbol{\gamma}_j(k)] \quad (10)$$

$$\boldsymbol{\beta}_{jf}(k+1) = \mathcal{T}_{\mu/J}\left(c\boldsymbol{\gamma}_{jf}(k) - \mathbf{p}_{jf}(k) - \mathbf{v}_{jf}(k) + c\sum_{j'\in\mathcal{N}_j}[\boldsymbol{\beta}_{jf}(k) + \boldsymbol{\beta}_{j'f}(k)]\right)/[c(2|\mathcal{N}_j| + 1)], \quad (11)$$

$$\boldsymbol{\gamma}_j(k+1) = \mathbf{M}_j^{-1}\left(\mathbf{X}_j'\mathbf{y}_j + c\boldsymbol{\beta}_j(k+1) + \mathbf{v}_j(k)\right). \quad (12)$$

Recursions (9)-(12) comprise the novel DGLasso algorithm, tabulated as Algorithm 1.

The algorithm entails the following steps. During iteration $k+1$, agent j receives the local estimates $\{\boldsymbol{\beta}_{j'}(k)\}_{j'\in\mathcal{N}_j}$ from the neighboring agents and plugs them into (9) to evaluate the dual price vector $\mathbf{p}_j(k)$. The new multiplier $\mathbf{v}_j(k)$ is then obtained using the locally available vectors $\{\boldsymbol{\gamma}_j(k), \boldsymbol{\beta}_j(k)\}$. Subsequently, vectors $\{\mathbf{p}_j(k), \mathbf{v}_j(k)\}$ are jointly used along with $\{\boldsymbol{\beta}_{j'}(k)\}_{j'\in\mathcal{N}_j}$ to obtain $\boldsymbol{\beta}_j(k+1)$ via N_f parallel vector soft-thresholding operations $\mathcal{T}_{\mu/J}(\cdot)$ defined in (6). Finally, the updated $\boldsymbol{\gamma}_j(k+1)$ is obtained from (12), and requires the previously updated quantities along with the vector of local observations $\mathbf{y}_j$ and regression matrix $\mathbf{X}_j$. The $(k+1)$st iteration is concluded after agent j broadcasts $\boldsymbol{\beta}_j(k+1)$ to its neighbors. The distributed $K-$fold CV protocol in [13] can be utilized to tune μ.

DGLasso algorithm does not require nested iterations, since all local updates are given in closed form. Even if an

arbitrary initialization is allowed, the sparse nature of the estimator sought suggest the all-zero vectors as a natural choice. With regards to communication cost, only the p scalars in $\boldsymbol{\beta}_j$ have to be broadcasted per iteration. When p is large, major savings can be attained by only exchanging the set of nonzero entries. Further, the inter-agent communication cost does not depend on the size of the local training sets. A computational cost analysis will be deferred to the ensuing section.

Remark 1 *(Reduction to distributed Lasso algorithm)* When $N_f = p$ and there are as many groups as entries of $\boldsymbol{\beta}$, then the sum $\sum_{f=1}^{N_f} \|\boldsymbol{\beta}_f\|$ becomes the ℓ_1-norm of $\boldsymbol{\beta}$, and group-Lasso reduces to Lasso. In this case, DGLasso offers a distributed algorithm to solve Lasso that coincides with the one in [13].

To close this section, it is useful to mention that convergence of Algorithm 1 is ensured by the convergence of the AD-MoM [12]. This result is formally stated next.

Proposition 1: *Let $\mathcal{G}$ be a connected graph, and consider recursions* (9)-(12) *that comprise the DGLasso algorithm. Then, for any value of the step-size $c > 0$, the iterates $\boldsymbol{\beta}_j(k)$ converge to the group-Lasso solution [cf.* (2)*] as $k \to \infty$, i.e.,*

$$\lim_{k\to\infty} \boldsymbol{\beta}_j(k) = \hat{\boldsymbol{\beta}}_{glasso},\ \forall\, j \in \mathcal{J}. \tag{13}$$

In words, all local estimates $\boldsymbol{\beta}_j(k)$ achieve consensus asymptotically, converging to a common vector that coincides with the desired estimator $\hat{\boldsymbol{\beta}}_{\text{glasso}}$. Formally, if the number of parameters p exceeds the number of data n, then a unique solution of (1) is not guaranteed for a general design matrix $\mathbf{X}$. Proposition 1 remains valid however, if the right-hand side of (13) is replaced by the set of minima; i.e., $\lim_{k\to\infty} \boldsymbol{\beta}_j(k) \in \arg\min_{\boldsymbol{\beta}} \frac{1}{J}\sum_{j=1}^{J} \|\mathbf{y}_j - \mathbf{X}_j\boldsymbol{\beta}\|_2^2 + \mu \sum_{f=1}^{N_f} \|\boldsymbol{\beta}_f\|_2$.

From (13), all asymptotic (as n grows large) properties of centralized (group-)Lasso carry over to its distributed counterpart developed here. Those include not only the bias, but also weak support consistency as well as estimation consistency, which for the centralized (group-)Lasso have been studied in e.g., [14, 15]. One can for instance borrow the weighted versions of the sparsifying penalties in [14] and [15], with weights provided by the (distributed) LS estimates in order to ensure the estimators enjoy (asymptotically) the aforementioned *oracle properties*.

4. PARALLEL PROCESSING

The algorithmic framework developed so far for distributed sparse estimation, can also be applied to obtain efficient *centralized* (group-)Lasso solvers as special cases of Algorithm 1. These will be briefly described next, since they are important on their own right as standalone sparse linear regression tools. Moreover, they will serve as a baseline for comparison with the distributed algorithms of Section 3, for the purpose of establishing that DGLasso has the property of parallelizing computations in multiprocessor architectures.

Algorithm 2 : GLasso

Initialize to zero $\{\boldsymbol{\beta}(0), \boldsymbol{\gamma}(0), \mathbf{v}(-1)\}$, and run:
for $k = 0, 1, \ldots$ **do**
  Update $\mathbf{v}(k) = \mathbf{v}(k-1) + c[\boldsymbol{\beta}(k) - \boldsymbol{\gamma}(k)]$.
  Update $\boldsymbol{\beta}_f(k+1) = (1/c)\mathcal{T}_\mu\left(c\boldsymbol{\gamma}_f(k) - \mathbf{v}_f(k)\right),\ \forall\, f$.
  Update $\boldsymbol{\gamma}(k+1) = \mathbf{M}^{-1}\left(\mathbf{X}'\mathbf{y} + c\boldsymbol{\beta}(k+1) + \mathbf{v}(k)\right)$.
end for

Recalling the network setup described in Section 2, let $J = 1$ so that the network collapses to a single agent, and suppose that this central processing unit has available the training data set $\{\mathbf{y}, \mathbf{X}\}$, say. In this case DGLasso yields a novel algorithm for the standard (centralized) group-Lasso estimator (2), termed GLasso and tabulated as Algorithm 2. To arrive at this result, start from the DGLasso recursions (9)-(12) and note that: (i) index j can be dropped since there is a single agent; (ii) summations across neighborhoods disappear for the same reason; and (iii) $\mathbf{p}(k) = 0,\ \forall\, k$ since (9) simplifies to $\mathbf{p}(k) = \mathbf{p}(k-1)$ and $\mathbf{p}(-1) = 0$. Because there are no consensus constraints to be enforced, it is reasonable that $\mathbf{p}(k)$ is no longer needed. Alternatively, one can directly arrive at Algorithm 2 after applying AD-MoM iterations to solve the problem

$$\min_{\{\boldsymbol{\beta},\boldsymbol{\gamma}\}} \left[\frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\gamma}\|_2^2 + \mu \sum_{f=1}^{N_f} \|\boldsymbol{\beta}_f\|_2\right],\ \text{ s. t. } \boldsymbol{\gamma} = \boldsymbol{\beta} \tag{14}$$

which is equivalent to (1). The sequence of iterates $\boldsymbol{\beta}(k)$ generated by Algorithm 2 is thus provably convergent to $\hat{\boldsymbol{\beta}}_{\text{glasso}}$ as $k \to \infty$, for any value of $c > 0$ [12].

Notice that the thresholding operator $\mathcal{T}_\mu$ in GLasso sets the entire sub-vector $\boldsymbol{\beta}_f(k+1)$ to zero whenever $\|c\boldsymbol{\gamma}_f(k) - \mathbf{v}_f(k)\|_2$ does not exceed μ, in par with the group sparsifying property of group-Lasso. After the thresholding, a proportional shrinkage typical of ridge (ℓ_2-penalized) estimators is performed [2]. In this case however, the shrinkage by a factor of c is due to quadratic term in the augmented Lagrangian. Not surprisingly, thresholding/proportional shrinkage type of updates have been also obtained in cyclic coordinate descent (CD) algorithms for the elastic net [16]. Different from [1], GLasso can handle a general (not orthonormal) regression matrix $\mathbf{X}$. Compared to the block-CD method proposed in [4], GLasso does not require an inner Newton-Raphson recursion per iteration.

As discussed in Remark 1, if $N_f = p$ so that the factors coincide with the scalar entries of $\boldsymbol{\beta}$, then GLasso yields the Lasso estimator. In particular, the vector soft-thresholding operator simplifies to its well-known scalar counterpart $\mathcal{S}_\mu(z) := (|z| - \mu)_+ \text{sign}(z)$. The Lasso estimator is expressible in terms of $\mathcal{S}_\mu$ whenever the problem is orthonormal or scalar, and thus $\mathcal{S}_\mu$ typically characterizes the updates of cyclic CD solvers for Lasso [2, p. 93].

When specialized to Lasso, Algorithm 2 coincides with the split Bregman method in [5], provided a single iteration is carried out in the minimization of a suitable "energy" introduced in [5, eq. (1.1)]. Such inexact minimization heuristic is suggested in [5] for algorithmic efficiency reasons, without recognizing its relation to AD-MoM and lacking formal convergence guarantees. This connection between AD-MoM and the split Bregman method was also pointed out in [17]. To estimate hierarchical sparse models or reconstruct signals based on incomplete Fourier data, related ideas based on cost decoupling to capitalize on alternating minimization methods were applied in [18] and [19].

4.1. Computational load balancing

Consider the DGLasso recursions (9)-(12). Update (12) involves inversion of the $p \times p$ matrix $\mathbf{M}_j := c\mathbf{I}_p + \mathbf{X}_j'\mathbf{X}_j$ per agent, that may be computationally demanding for sufficiently large p. Fortunately, this operation as well as the evaluation of the local "ridge" estimate $[c\mathbf{I}_p+\mathbf{X}_j'\mathbf{X}_j]^{-1}\mathbf{X}_j'\mathbf{y}_j$ can be carried out offline before running the algorithm. Other than that, the updates comprising DGLasso are simple and solely involve scaling/addition and thresholding of (eventually sparse) p-dimensional vectors. As pointed out in [5], in several applications of interest there is specific structure that can be exploited to efficiently invert the aforementioned matrix. Circulant structure has been shown to arise in compressive sampling for magnetic resonance imaging [5], hence the inversion can be carried out through a suitable DFT. Sparsity is another characteristic of the matrix that can be capitalized upon in, e.g., multiple frequency-hopping signal estimation [20].

In any case, the matrix inversion lemma can be invoked to obtain

$$\mathbf{M}_j^{-1} = (1/c)\left[\mathbf{I}_p - \mathbf{X}_j'\left(c\mathbf{I}_{n_j} + \mathbf{X}_j\mathbf{X}_j'\right)^{-1}\mathbf{X}_j\right].$$

In this new form, the dimensionality of the matrix to invert becomes $n_j \times n_j$, where n_j is the number of locally acquired data. For highly underdetermined regression problems ($n_j \ll p$) typically arising in genomics or computational biology [2, Ch. 18], (D)GLasso enjoys considerable computational savings through the aforementioned matrix inversion identity. More importantly, one also recognizes that the distributed operation parallelizes the numerical computation across agents: if GLasso is run centrally with all network-wide data $\mathbf{y} := [\mathbf{y}_1', \ldots, \mathbf{y}_J']'$ and $\mathbf{X} := [\mathbf{X}_1', \ldots, \mathbf{X}_J']'$ at hand, then the matrix to invert has dimension $n = \sum_{j\in\mathcal{J}} n_j$, which increases linearly with the network size J. Beyond a networked scenario as described in Section 2, DGLasso provides an attractive alternative for computational load balancing in timely multi-processor architectures.

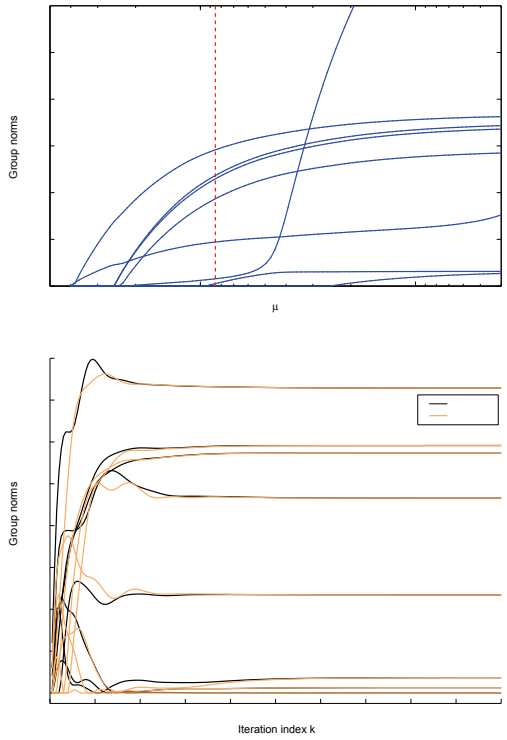

Fig. 1. (top) Group-Lasso regularization path; (bottom) evolution of the per factor norms for agents 2 and 7.

5. NUMERICAL EXAMPLE

A simulated test is now presented to corroborate the convergence of DGLasso. The example will rely on the birthweight dataset considered in the seminal group-Lasso work of [1]. The objective is to predict the human birthweight from $N_f = 8$ factors including the mother's `age`, `weight`, `race`, `smoke` habits, number of previous `premature` labors, history of `hypertension`, uterine `irritability`, and number of physician `visits` during the first trimester of pregnancy. Third-order polynomials were considered to model nonlinear effects of the `age` and `weight` on the response, augmenting the model size to $p = 12$ by grouping the polynomial coefficients in two subsets of three variables. The network of $J = 10$ agents is simulated as a random geometric graph on $[0, 1]^2$, with communication range $r = 0.4$. The $n = 189$ data samples are randomly split across agents, so that $n_j = 18$ for $j \in [1, 9]$, and $n_{10} = 27$.

By running Algorithm 1 and using "warm starts" [16], the path of group-Lasso solutions is computed at 100 different values of the regularization parameter μ. The penalty coefficient is set to $c = 8$, since several experiments suggested this value leads to fastest convergence. Fig. 1 (top) shows the regularization path for agent $j = 2$, where for diminishing values of μ more factors enter the model. The dashed vertical line indicates the model for $\mu_{\text{CV}} = 8.513$, obtained via the 10-fold distributed CV procedure in [13]. Consensus is

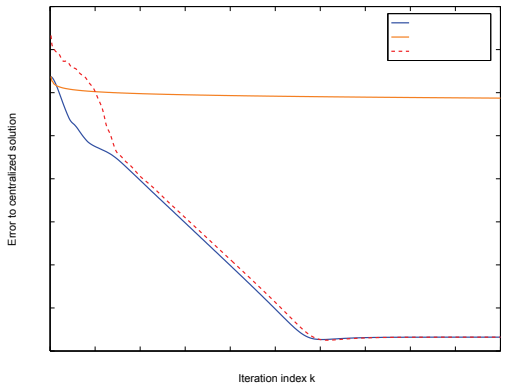

Fig. 2. Global estimation error evolution.

achieved after few iterations, as observed from Fig. 1 (bottom) which depicts the evolution of the factors' strength measured by $\|\boldsymbol{\beta}_{jf}\|_2$, for two representative agents with $j = 2, 7$. DGLasso converges to the same prediction model as in [1], and determines that `visits` is not significant even from the first iterations, allowing for early model selection.

We also compare DGLasso with the distributed subgradient method in [11], for which equal neighbor combining weights, zero initial conditions, and a diminishing stepsize $\alpha(k) = 10^{-2}/k$ are adopted. As figure of merit, the global error metric $\epsilon(k) := J^{-1} \sum_{j=1}^{J} \|\hat{\boldsymbol{\beta}}_j(k) - \hat{\boldsymbol{\beta}}_{\text{glasso}}\|_2^2$ is evaluated for all schemes. Algorithm 2 was utilized to obtain $\hat{\boldsymbol{\beta}}_{\text{glasso}}$, and the resulting errors are depicted in Fig. 2. The decreasing trend of $\epsilon(k)$ confirms that all local estimates converge to $\hat{\boldsymbol{\beta}}_{\text{glasso}}$, as stated in Proposition 1. All-zero initial vectors speed up DGLasso. With regards to the subgradient method, the speed of convergence is extremely slow since a descent along a subgradient direction is not effective in nulling factors of the local estimates.

6. CONCLUDING REMARKS

An in-network processing-based algorithm for fitting a group-Lasso model is developed in this paper, based on AD-MoM iterations. Apart from an offline matrix inversion, the resulting per agent DGLasso updates are simple and given in closed form. In a nutshell, the DGLasso recursions entail linear combinations of vectors and a soft-thresholding operation. Interestingly, DGLasso has the property of parallelizing computations across agents, and requires affordable communications of sparse messages within the neighborhood. The sequences of local estimates generated by DGLasso are provably convergent to $\hat{\boldsymbol{\beta}}_{\text{glasso}}$, and the algorithm can outperform alternatives based on distributed subgradient descent.

7. REFERENCES

[1] M. Yuan and Y.Lin, "Model selection and estimation in regression with grouped variables," *J. Royal Statistic. Soc. B*, vol. 68, pp. 49–67, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, second edition, 2009.

[3] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal. Statist. Soc. B*, vol. 58, pp. 267–288, 1996.

[4] A. T. Puig, A. Wiesel, and A. O. Hero, "A multidimensional shrinkage-thresholding operator," in *Proc. of Wkshp. on Stat. Signal. Proc.*, Cardiff, Wales, Aug/Sep 2009.

[5] T. Goldstein and S. Osher, "The split bregman method for l1 regularized problems," *SIAM Journal on Imaging Sciences*, vol. 2, pp. 323–343, 2009.

[6] J. Friedman, T. Hastie, and R. Tibshirani, "A note on the group lasso and sparse group lasso," Technical Report, 2010.

[7] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Trans. Signal Process.*, vol. 57, pp. 2479–2493, 2009.

[8] V. Kekatos and G. B. Giannakis, "Selecting reliable sensors via convex optimization," in *Proc. of Intl. Wkshp. on Signal Proc. Adv. in Wireless Comm.*, Marrakech, Morroco, June 2010.

[9] J.-A Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Trans. Signal Process.*, vol. 58, pp. 1847–1862, 2010.

[10] J.-A Bazerque, G. Mateos, and G. B. Giannakis, "Group-lasso on splines for spectrum cartography," *IEEE Trans. Signal Process.*, (revised).

[11] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, pp. 48–61, 2009.

[12] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena-Scientific, 1999.

[13] G. Mateos, J.-A Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, 2010.

[14] H. Zou, "The adaptive lasso and its oracle properties," *J. Amer. Statist. Assoc.*, vol. 101, pp. 1418–1429, 2006.

[15] H. Wang and C. Leng, "A note on adaptive group lasso," *Computational Statistics and Data Analysis*, vol. 52, pp. 5277–5286, 2008.

[16] J. Friedman, T. Hastie, and R. Tibshirani, "Regularized paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, 2010.

[17] E. Esser, "Applications of lagrangian-based alternating direction methods and connections to split bregman," Technical Report, 2009.

[18] P. Sprechmann, I. Ramirez, G. Sapiro, and Y. C. Eldar, "Collaborative hierarchical sparse modeling," in *Proc. of 44th Conf. on Info. Sciences and Systems*, Princeton, NJ, March 2010.

[19] J. Yang, Y. Zhang, and W. Yin, "A fast alternating direction method for TVL1-L2 signal reconstruction from partial fourier data," *IEEE Jrnl. Sel. Topics in Signal Process.*, vol. 4, pp. 288–297, 2010.

[20] D. Angelosante, G. B. Giannakis, and N. D. Sidiropoulos, "Estimating multiple frequency-hopping signal parameters via sparse linear regression," *IEEE Trans. Signal Process.*, 2010.