Distributed Recovery of Jointly Sparse Signals Under Communication Constraints

Distributed recovery of jointly sparse signals under communication constraints

Sophie M. Fosson* Javier Matamoros† Carles Antón-Haro† Enrico Magli*
* Department of Electronics and Telecommunications, Politecnico di Torino (Italy)
†Centre Tecnològic de Telecomunicacions de Catalunya, Barcelona (Spain)

***Abstract*—The problem of the distributed recovery of jointly sparse signals has attracted much attention recently. Let us assume that the nodes of a network observe different sparse signals with common support; starting from linear, compressed measurements, and exploiting network communication, each node aims at reconstructing the support and the non-zero values of its observed signal. In the literature, distributed greedy algorithms have been proposed to tackle this problem, among which the most reliable ones require a large amount of transmitted data, which barely adapts to realistic network communication constraints. In this work, we address the problem through a reweighted ℓ_1 soft thresholding technique, in which the threshold is iteratively tuned based on the current estimate of the support. The proposed method adapts to constrained networks, as it requires only local communication among neighbors, and the transmitted messages are indices from a finite set. We analytically prove the convergence of the proposed algorithm and we show that it outperforms the state-of-the-art greedy methods in terms of balance between recovery accuracy and communication load.**

***Index Terms*—Joint sparsity, distributed algorithms, compressed sensing, iterative thresholding, reweighted ℓ_1 minimization, concave penalization.**

I. Introduction

The recovery of jointly sparse signals has received great attention in the last few years. By "jointly sparse" we mean signals that are sparse (*i.e.*, have few non-zero components) with same support (*i.e.*, the positions of the non-zero components are common for all the signals). Measurements of such signals are assumed to be taken by the nodes of a network; given the measurements, the aim of each node is to estimate the common support and eventually evaluate the non-zero components. The study of this problem is motivated by diverse applications, among which one of the most outstanding is spectrum sensing in cognitive radio networks [1], [2], which consists in the detection of the spectrum occupancy aimed to the dynamic reallocation of unused frequencies; as described in [2, Section III.D], in some cases this problem reduces to the reconstruction of a common support. Other examples of jointly sparse representations, just to name a few among the most recent ones, arise from image features extraction [3], visual classification [4], speech recognition [5], and biometrics recognition [6].

In several applications, measurements are linearly acquired and compressed [1], [2], according to the distributed compressed sensing (CS) paradigm [7], [8]. CS [9] states that a sparse signal $x \in \mathbb{R}^n$ can be recovered from measurements $y = Ax$ where $A \in \mathbb{R}^{m,n}$ is a suitable matrix with $m < n$, called sensing matrix. In a distributed context, the acquisition is performed by a networked system: given a set $\mathcal{V}$ of nodes, each $v \in \mathcal{V}$ has its own measurement $y_v = A_v x_v$; the case when the x_v's have common support is known as joint sparsity model 2 (JSM-2, [8]). Concerning the recovery methods, centralized and distributed methods have to be distinguished. The first ones assume the presence of a fusion center that gathers all the information from the network (namely, measurements and sensing matrices) and processes them to recover the signals. In the case that all the sensing matrices are equal, these methods can be recast in the multiple measurement vectors framework (MMV) [10], for which theoretical recovery guarantees have been provided [10], [11]. More insight on the recovery methods for MMV can be found in very recent papers such as [12], [13]. The distributed recovery methods, instead, perform the reconstruction in-network, with no fusion center, only exploiting the computational and (local) communication capabilities of the nodes. Distributed methods are remarkable as (a) they do not need the presence of a fusion center, which in many situations is not available or can be expensive to reach in terms of transmit power (sensor networks are often deployed over impracticable territories for environment monitoring purposes); (b) they are more robust to failures: if a fusion center breaks down, the recovery process stops, while if a distributed algorithm is run in-network, typically the failure of some nodes is tolerated.

The development of distributed recovery algorithms for JSM-2 is our purpose. The literature on this argument is very recent. First attempts [14], [2, Section III.D] went in the direction of decentralizing group Lasso techniques [15], but no convergence guarantees were provided. Distributed greedy algorithms were then studied: in [16], distributed versions of subspace pursuit (SP) and orthogonal matching pursuit (OMP) were developed, the second one (called DiOMP) being more promising in terms of recovery performance. The support recovery accuracy of DiOMP is comparable to that of DiT in [17], which is the first distributed algorithm based on iterative thresholding for JSM-2. Almost at the same time, in [18] DCOMP 1 was proposed, which is very similar to DiOMP, but

This work is supported by the European Commission in the framework of the FP7 Network of Excellence in Wireless COMmunications, Grant agreement n.318306, by the European Research Council under FP7 / ERC, Grant agreement n.279848 - CRISP project, by the Spanish Government through the project INTENSYV (TEC2013-44591-P), and by the Catalan Government (2014 SGR 1567).

more accurate in the support detection. A second algorithm was proposed in [18], named DC-OMP 2, which recovers the support much more accurately than DC-OMP 1, at the price of a greater communication load. To the best of our knowledge, DC-OMP 1 and DC-OMP 2 represent the state of the art in the framework of distributed algorithms for JSM-2 and will be considered as benchmark in this work; in the following, we will describe them more in detail. The aim of this paper is to present a new approach to the distributed recovery of jointly sparse signals, based on concave penalization and reweighted ℓ_1 minimization. More precisely, we will develop a distributed soft thresholding in which the threshold is iteratively updated, based on the support estimate. With our method, communication can be strongly reduced with respect to DC-OMP 2 (with no performance loss), being limited to the local communication of the indices of the components that have switched from non-zero to zero or vice versa. In other terms, our algorithm will be efficient even under strict communication constraints, due to the network technology or for energy saving purposes. Our algorithm will be proved to converge to a minimum of suitable cost functional, and performance will be shown via numerical simulations.

The paper is organized as follows. In Section II, we will describe the model, and in Section III we will establish our optimization problem. In Section IV, we will present and discuss our algorithm. In Section V-A we will prove the numerical convergence and the stabilization of the support estimate, while the convergence of the non-zero components will be discussed in Section V-B. Numerical results will be then shown in Section VI, along with an analysis of the transmission costs. Finally, some conclusions will be drawn.

Before proceeding, we anticipate some notation that will be used throughout the paper.

A. Notation

We denote by $\mathbb{1}$ the indicator function: for any integer $n \geq 1$, $\mathbb{1} : \mathbb{R}^n \mapsto \mathbb{R}^n$ is given by $[\mathbb{1}(x)]_i = 1$ if $x_i \neq 0$, while $[\mathbb{1}(x)]_i = 0$ if $x_i = 0$, $i = 1, \ldots, n$. $\mathbf{1}$ indicates the column vector whose components are all equal to 1. We define the l_0-norm of a vector $x \in \mathbb{R}^n$ as $\|x\|_0 = \|\mathbb{1}(x)\|_2^2$, or equivalently $\|x\|_0 = \mathbf{1}^\mathsf{T}\mathbb{1}(x)$, where T indicates the transpose. I is the identity matrix. Moreover, we call weighted l_p-norm of x the quantity $\|Wx\|_p$ where W is a weight matrix, namely a diagonal matrix with diagonal entries $W_i > 0$, $i = 1, \ldots, n$. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for any node $v \in \mathcal{V}$, $\mathcal{N}_v := \{w \in \mathcal{V} \text{ s.t. } (v, w) \in \mathcal{E}\}$ is the neighborhood of v. Let d_v be the degree of v, say the number of neighbors of v, included v itself. Given any variable x_v associated with v, we indicate its local average with an overline: $\overline{x}_v := \frac{1}{d_v} \sum_{w \in \mathcal{N}_w} x_w$ (we remark that $:=$ denotes "is defined as").

II. Network model

In this section, we describe the acquisition and communication model of interest.

We consider a network composed of V nodes, whose connectivity is described by the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = V$. Accordingly, the node v can communicate with v' if and only if $\{v, v'\} \in \mathcal{E}$ or, in other words, if v' belongs to its neighborhood set $\mathcal{N}_v$.

Following the CS paradigm, each node observes a compressed version of a k-sparse signal $\{x_v^\star\}_{v \in \mathcal{V}} \in \mathbb{R}^n$ through a set of linear measurements, namely

$$y_v = A_v x_v^\star, \quad v \in \mathcal{V} \tag{1}$$

where $A_v \in \mathbb{R}^{m \times n}$ (with $m < n$) and the signals $\{x_v\}_{v \in \mathcal{V}}$ have the same support Ω, that is, for all $v \in \mathcal{V}$, $\Omega_v := \left\{i \in \{1, \ldots, n\} | x_{v,i}^\star \neq 0\right\} = \Omega$. In the next, we will equivalently refer to the support of x_v as the binary vector $\mathbb{1}(x_v)$.

A measurement noise term can be added in (1) to have a more realistic setting. If we assume an additive white Gaussian noise (a popular choice in a number of applications), the formulation and the approach to the problem do not change with respect to the noiseless case, as we consider the least squares paradigm, which in both cases considers the minimization of the residual.

The ultimate goal of each node $v \in \mathcal{V}$ is the reconstruction of its observed signal x_v. A fusion center is not envisaged in our model, thus the reconstruction task has to be performed in-network by the nodes themselves. Moreover, we assume that no information about A_v and y_v can be shared, *e.g.*, for privacy reasons and to reduce the amount of transmitted data. Since the transmission load is often a dramatic drawback in distributed procedures, we impose a second constraint on the communication protocol: messages must belong to a finite set of integers, specifically $\{1, \ldots, n\}$. This should adapt to our purpose: since the support is the common quantity, it should be sufficient to share information about the support of each component, which is a binary message. In other terms, for each component i a node would communicate its status, that is, if in its current estimate i is in the support or not; assuming that the other nodes can store such information, it is sufficient to send the value i when the status has changed. For each sent message, we then need only $\lfloor \log_2 n \rfloor + 1$ bits, which generally is significantly smaller than the number of bits used to transmit a real number, even if coarsely quantized.

Let us summarize these communication constraints.

Assumption 1. *The communication over the network is local, and only messages in $\{1, \ldots, n\}$ can be transmitted by each node to the neighbors.*

It is well known that, in the CS context, the challenge is the identification of the signal support; once this is done, the estimate of the non-zero components could be readily performed through the classical least squares estimation (assumed the number of measurements is larger than the sparsity). For this motivation, in the literature [16], [18] the detection of the signal support is approached separately. Our proposed method instead will envisage both support and non-zero values recovery in the same algorithm.

III. Optimization problem

Given the network model presented in Section II, we now describe our recovery problem in terms of an optimization problem, that takes into account the network constraints of

Assumption 1. Our final purpose is the development of a distributed recovery algorithm that leverages iterated sharing of information about the support.

In the context of sparse recovery, the ℓ_1 convex minimization problem, known as Lasso, is very popular for its mathematical feasibility. The principle behind Lasso is that ℓ_1 norm well approximates the ℓ_0 norm and allows to transform the recovery problem into a convex problem. Further, *reweighted ℓ_1 minimization* [19], [20], [21] has been proposed, which iteratively retunes the weight of the ℓ_1 norm based on the current signal's estimate. In this way, each component is weighted according to its expectation of belonging to the support. Different reweighting rules have been investigated in the literature, and will be discussed later.

The reweighting principle seems to be suitable for distributed support detection: intuitively we can think of an ℓ_1-reweighting minimization at each node, in which the reweighting rule depends on the individual current estimate and on the support information shared in the network. In other terms, we aim for a decentralization of reweighted ℓ_1 minimization.

The rest of the section is devoted to develop this idea. We start with a review on (centralized) *concave penalization*, which is the setting where the ℓ_1 reweighting techniques are originated. Afterwards, we will illustrate how to decentralize this method, taking into account our model constraints (Assumption 1).

A. From Lasso to concave penalization

As mentioned before, the problem of sparse signals' recovery can be conceived as an ℓ_1 convex minimization problem, known as Lasso:

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|y-Ax\|_2^2 + \lambda\|x\|_1, \quad \lambda>0 \tag{2}$$

where $A\in\mathbb{R}^{m\times n}$, and λ is a parameter to set. As already said, the ℓ_1 norm has been shown to well approximate the ℓ_0 norm, and has the great advantage of transforming the problem from combinatorial to convex. However, Lasso has some drawbacks, namely its estimate is always biased (proportionally to λ), and conditions to have the oracle property (*i.e.*, the capability of exactly recovering the support) are strict [22], [23], [24]. This has motivated the studies on different penalization techniques. In particular, much interest has been devoted to concave penalization techniques:

$$\begin{aligned}&\min_{x\in\mathbb{R}^n} \frac{1}{2}\|y-Ax\|_2^2 + \lambda\sum_{i=1}^n g(|x_i|)\\ &g:\mathbb{R}_+\to\mathbb{R}_+ \text{ concave, nondecreasing in } |x_i|.\end{aligned} \tag{3}$$

The rationale behind this is that concave functions approximate the ℓ_0 norm better than ℓ_1, as one can appreciate in Figure 1. Many contributions on concave penalization come from the statistical community, see, *e.g.*, [22], [25], [26], [27], [28], [29], [30]. In such papers, different concave g's have been proposed, and conditions to have the oracle property and to reduce the Lasso bias have been studied, mainly in the asymptotic case $n\to\infty$ [22], [25]. Experimental and theoretical results attest that usually concave penalization outperforms

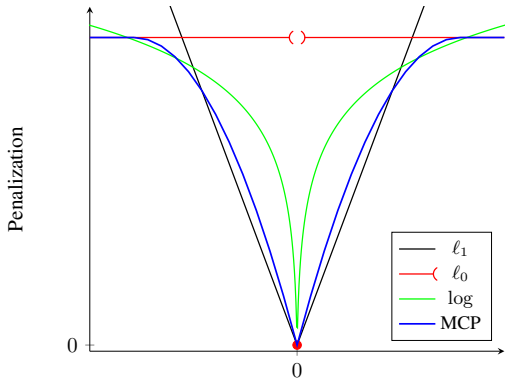


Figure 1: Examples of popular concave penalization functions, that are closer to ℓ_0 than ℓ_1. In this work, we focus on MCP.

Lasso [22], [28], [29]. In the context of underdetermined linear systems, some works [19], [31] apply the concave penalization to CS and matrix rank minimization with success.

The concave penalization problem (3) is not mathematically straightforward: non-convexity makes it difficult to find global solutions. However, in many cases local minima are precise enough, and can be reached via iterative methods based on *linear local approximation* (LLA) of g [27], [19], [31]. Given a point $z_i\in\mathbb{R}_+$, the key idea of LLA is to substitute $g(|x_i|)$ around z_i by its linearization $g(z_i)+g'(z_i)(|x_i|-z_i)$; thanks to concavity, g is always below its linearization, which suggests the following procedure. Assuming that z is the current estimate, we locally minimize (3) substituting g with its linearization. Removing the constant terms, we obtain:

$$\min_{x\in\mathbb{R}^n} \frac{1}{2}\|y-Ax\|_2^2 + \lambda\sum_{i=1}^n g'(|z_i|)|x_i|. \tag{4}$$

Let us suppose that an estimate $z=x(t)$ is provided at current time $t\in\mathbb{N}$. Then, we can perform alternated minimization on (4):

$$\begin{aligned}x(t+1) &= \min_{x\in\mathbb{R}^n} \frac{1}{2}\|y-Ax\|_2^2 + \lambda\sum_{i=1}^n w_i(t)|x_i|\\ w_i(t+1) &= g'(|x_i(t+1)|).\end{aligned} \tag{5}$$

This turns out to be is an iterative reweighted ℓ_1 minimization procedure. Such method has been proved to reach a local minimum of the concave penalization functional, and in practice it is more accurate than Lasso global solution [27], [19], [31]. We remark that no general guarantee of convergence for $x_i(t)$ is provided, but specific results hold for specific g's. For example in [21], convergence is proved for $g(|x_i|)=(|x_i|+\epsilon)^p$, with $p\in(0,1)$ and small $\epsilon>0$.

In the literature, a variety of concave penalization functions have been investigated. In [19] much attention is focused on the case $g(|x_i|)=\log(|x_i|+\epsilon)$, with small $\epsilon>0$. SCAD [22] and MCP [28] instead propose continuous quadratic penalizations: MCP is of the form $g(x_i)=\alpha|x_i|-\beta x_i^2$, for $|x_i|<\frac{\alpha}{2\beta}$, $\alpha,\beta>0$, and constant otherwise; SCAD is like MCP plus a ℓ_1 penalization term $\lambda|x_i|$ for small $|x_i|$. In the cited works, in-depth analyses and comparisons between the different g's are proposed.

In conclusion, concave penalization provides us (a) a sparse recovery setting that outperforms Lasso, and (b) low complex algorithms, based on LLA, to find a solution. The LLA algorithms are nothing but reweighted ℓ_1 schemes.

B. Decentralization under communication constraints

Our aim is to decentralize the problem (3) and the algorithm sketched by (4)-(5) under communication constraints (Assumption 1). First of all, we notice that the natural way to write the optimization problem over the network is the summation of the individual functionals (3) for each node $v \in \mathcal{V}$. Second, we observe that the penalization is strictly linked to the support: as explained in [19], in (4) we would desire larger w_i's for the zero components, up to the ideal case when $w_i \to \infty$ for zero components, and $w_i \to 0$ for non-zero components. Since here signals have common support, it makes sense to compute g over a common variable of the network, and the simplest choice is the mean. Summing up, we have:

$$\min_{x_v \in \mathbb{R}^n} \sum_{v \in \mathcal{V}} \left\{ \frac{1}{2} \left\| y_v - A_v x_v \right\|_2^2 + \lambda \sum_{i=1}^{n} g\left(\frac{1}{V} \sum_{v \in \mathcal{V}} |x_{v,i}| \right) \right\}.$$

Nevertheless, this would require global communication to update w in the procedure (5), which is in contrast with Assumption 1 for non-complete graphs. We then use the best local approximation that we can conceive, that is, we substitute $\frac{1}{V}\sum_{v\in\mathcal{V}} |x_{v,i}|$ with the local sum $\frac{1}{|\mathcal{N}_v|}\sum_{u\in\mathcal{N}_v} |x_{u,i}|$. The corresponding functional is

$$\min_{x_v \in \mathbb{R}^n} \sum_{v \in \mathcal{V}} \left\{ \frac{1}{2} \left\| y_v - A_v x_v \right\|_2^2 + \lambda \sum_{i=1}^{n} g\left(\frac{1}{|\mathcal{N}_v|} \sum_{u \in \mathcal{N}_v} |x_{u,i}| \right) \right\}.$$

In this way, each $v \in \mathcal{V}$ will have its own weight w_v, which will be reweighted using only local collaboration.

Finally, according to Assumption 1 the transmission of real valued messages (such as $|x_{u,i}|$) is undesired. Therefore, we impose that each node v cannot access x_u, $u \in \mathcal{N}_v \setminus \{v\}$, but only their best "binary approximation", say $\mathbb{1}(x_{u,i}(t))$. We then substitute $|x_{u,i}|$ by $\mathbb{1}(x_{u,i})$, and obtain our ultimate minimization problem: given $X = (x_1, \ldots, x_V)$, we write

$$\min_{x_v \in \mathbb{R}^n} \mathcal{F}(X) \tag{6}$$

where

$$\mathcal{F}(X) = \sum_{v \in \mathcal{V}} \left\{ \frac{1}{2} \left\| y_v - A_v x_v \right\|_2^2 + \lambda \sum_{i=1}^{n} g\left(\alpha |x_{v,i}| + \overline{\mathbb{1}(x_{v,i})} \right) \right\}$$

and $\overline{\mathbb{1}(x_{v,i})} = \frac{1}{|\mathcal{N}_v \setminus v|} \sum_{u \in \mathcal{N}_v \setminus v} \mathbb{1}(x_{u,i})$[1]. $\alpha > 0$ is a tuning parameter: since we are summing quantities that are physically different (a magnitude $|x_{v,i}|$ and binary information), it could be useful to balance their contributions, *e.g.* based on prior information on the energy of the signal. In practice, we have noticed that if each v adds also $\mathbb{1}(x_{v,i})$, performance improves; therefore, in the following we will use $\overline{\mathbb{1}(x_{v,i})} = \frac{1}{|\mathcal{N}_v|} \sum_{u \in \mathcal{N}_v} (\mathbb{1}(x_{u,i}))$.

[1]We remark that $\mathbb{1}(x_{u,i})$ is a function of $|x_{u,i}|$, which guarantees that the current $g = g\left(\alpha|x_{v,i}| + \overline{\mathbb{1}(x_{v,i})}\right)$ is still a function of the absolute values.

Summing up, the LLA procedure applied to $\mathcal{F}(X)$ originates the following decentralized reweighted ℓ_1 minimization procedure:

$$\begin{aligned} x_v(t+1) &= \min_{x_v \in \mathbb{R}^n} \mathcal{F}_w(X) \\ w_{v,i}(t+1) &= g'\left(\alpha|x_{v,i}(t+1)| + \overline{\mathbb{1}(x_{v,i}(t+1))}\right) \end{aligned} \tag{7}$$

where $\mathcal{F}_w(X)$ is $\mathcal{F}(X)$ with $w_{v,i}(t)\left[\alpha|x_{v,i}| + \overline{\mathbb{1}(x_{v,i})}\right]$ instead of $g\left(\alpha|x_{v,i}| + \overline{\mathbb{1}(x_{v,i})}\right)$.

Assuming that each v can store n bits for each one of its neighbors, the neighbors are just required to broadcast the message i when the status (0 or 1) of the component i has changed in the current estimation, which fulfills Assumption 1.

Concerning the update of x_v in (7), three tricky points arise and will be discussed in next section. The minimization of $\mathcal{F}_w(X)$ over x_v:

1) is not a classical Lasso minimization due to the presence of the terms $\overline{\mathbb{1}(x_v)}$;
2) requires the local communication of the w_v's, which is still in contrast with Assumption 1;
3) is too fast for our networked problem: we observed in fact that the whole procedure converges after few iterations. This is undesirable because it does not allow propagation of the information over the network. We will then make the procedure slower by not computing the minimum, but just decreasing $\mathcal{F}$ with respect to x_v, via an iterative thresholding step.

Before proceeding, we specify that in this work we will focus on the following concave penalization function g:

$$g(|z|) = \begin{cases} \beta|z| - \frac{1}{2}z^2 & \text{if } 0 \le z < \beta \\ \frac{1}{2}\beta^2 & \text{otherwise.} \end{cases} \qquad z \in \mathbb{R}, \beta > 0 \tag{8}$$

This g belongs to the family of MCP penalization functions [28], and has been recently exploited in applications such as wavelets [32, Equation 2.8] and Gaussian Bayesian networks [33]. As explained in [28], MCP is appreciated as it minimizes the maximum concavity. In Figure 1 we compare g in (8) to other classical choices. Notice that when $|z| \ge \beta$, g is constant and more penalization is applied, hence β is a penalization threshold that can be tuned based on the problem. With (8), in (7) we have:

$$w_{v,i}(t) = [\beta - \alpha|x_{v,i}(t)| - \overline{\mathbb{1}(x_{v,i}(t)}]_+ \tag{9}$$

where $[z]_+ = \max\{0, z\}$, $z \in \mathbb{R}$.

The motivation to focus on (8) is twofold: on one hand, experimental results are satisfactory (see Section VI); on the other hand, the mathematical simplicity of (8) allows us to provide a complete convergence analysis of $x_v(t)$ (see Section V-B). In the next section, we discuss the update of $x_v(t)$ using this g, and we finally state our algorithm.

IV. Proposed algorithm

Let us tackle points 1), 2), and 3) underlined in the previous section, that complicate algorithm (7). First of all, let us notice

that we can separate the terms of $\mathcal{F}_w(X)$ that depend on single x_v's, and we indicate them by $\mathcal{F}_w(x_v)$:

$$\begin{aligned}\mathcal{F}_w(x_v) =&\frac{1}{2}\left\|y - A_v x_v\right\|_2^2 + \lambda \sum_{i=1}^{n} w_{v,i}\alpha|x_{v,i}| \\ &+ \lambda \sum_{i=1}^{n} \mathbb{1}(x_{v,i}) \sum_{u\in\mathcal{N}_v} \frac{w_{v,i}}{|\mathcal{N}_u|}.\end{aligned} \tag{10}$$

This formula highlights that each $v \in \mathcal{V}$ has to solve a Lasso with an extra term, *i.e.*, a weighted ℓ_0 norm, as anticipated in point 1), Section III. In other terms, $\mathcal{F}_w(x_v)$ has both ℓ_1 and ℓ_0 penalizations. Moreover, point 2) is now evident: the transmission of the neighboring w_u's is necessary to compute $\mathbb{1}(x_{v,i})\sum_{u\in\mathcal{N}_v}\frac{w_{v,i}}{|\mathcal{N}_u|}$. In the next, we will use the notation $\overline{w_{v,i}} = \mathbb{1}(x_{v,i})\sum_{u\in\mathcal{N}_v}\frac{w_{v,i}}{|\mathcal{N}_u|}$.

In order to face point 3), we replace the minimization step with a decreasing step, that slows down the algorithm's convergence. Given the shape of $\mathcal{F}_w(x_v)$, iterative thresholding is a suitable choice for this purpose. In [34, Section 4.1], the *soft* thresholding algorithm has been proved to decrease the Lasso functional [34, Lemma 4.3] by showing that it iteratively minimizes a properly augmented functional, known as surrogate functional. A similar property has been proved also for the *hard* thresholding algorithm in [35], which decreases the ℓ_0 penalized functional. Here, we use the same scheme based on the surrogate functional to develop an iterative thresholding algorithm that decreases $\mathcal{F}$. Due to the presence of both ℓ_1 and ℓ_0 terms, such procedure will merge soft and hard features. We refer the interested reader to [36], [37] and to [12] for a deeper insight into hard and soft/hard thresholding techniques, respectively.

We remark that efficient methods like the alternating direction method of multipliers (ADMM), [38], [39] cannot be directly implemented due to the non-convexity of $\mathcal{F}$. This will be further elaborated in Sections IV-A and VI-F. On the other hand, in the literature algorithms for the minimization of non-convex, non-smooth problems have been recently presented [40], [41], [42], [43], which here cannot be applied due to the non-continuity of $\mathcal{F}$.

Let $B = (b_1, \dots, b_V) \in \mathbb{R}^{n\times V}$. We define the surrogate functional as follows (see [34, Section 4.1.1] and [35, Section 2.2]):

$$\mathcal{R}(X,B) := \mathcal{F}(X) + \frac{1}{2}\sum_{v\in\mathcal{V}}\left[\frac{1}{\tau}\left\|x_v - b_v\right\|_2^2 - \left\|A_v(x_v - b_v)\right\|_2^2\right].$$

By defining $z_v := b_v + \tau A_v^{\mathsf{T}}(y_v - A_v b_v)$, the following equality can be readily proved ([34, Section 4.1.1]):

$$\begin{aligned}&\left\|y_v - A_v x_v\right\|_2^2 + \frac{1}{\tau}\left\|x_v - b_v\right\|_2^2 - \left\|A_v(x_v - b_v)\right\|_2^2 = \\ &= \frac{1}{\tau}\left\|x_v - z_v\right\|_2^2 + \mathsf{const}\end{aligned}$$

where const is a term not depending on x_v. Hence, we can write the surrogate of each $\mathcal{F}_w(x_v)$ as:

$$\mathcal{R}_w(x_{v,i}) = \frac{1}{2\tau}(x_{v,i} - z_{v,i})^2 + \lambda\left[\alpha w_{v,i}|x_{v,i}| + \mathbb{1}(x_{v,i})\overline{w_{v,i}}\right]. \tag{11}$$

Following the procedure in [34, Section 4.1.1], we minimize $\mathcal{R}_w(x_{v,i})$ in (11) with respect to $x_{v,i}$. We distinguish two cases.

1) $|z_{v,i}| \le w_{v,i}$: $\operatorname{argmin}\mathcal{R}(x_{v,i}) = 0$.

   In fact, if $|z_{v,i}| \le w_{v,i}$ and $x_{v,i} \neq 0$, the derivative of $\mathcal{R}_w(x_{v,i})$ is $x_{v,i} - z_{v,i} + \operatorname{sgn}(x_{v,i})w$, which is positive for $x_{v,i} > 0$, and symmetrically negative for $x_{v,i} < 0$. We then have the infimum points $\lim_{x_{v,i}\to 0+}\mathcal{R}_{x_{v,i}} = \frac{1}{2\tau}z_{v,i}^2 + \lambda\overline{w_{v,i}} \ge \frac{1}{2\tau}z_{v,i}^2 = \mathcal{R}_w(0)$, which shows that the global minimum is in zero, as depicted in Figure 2.(a).
2) $|z_{v,i}| > w_{v,i}$: if $(|z_{v,i}| - w_{v,i})^2 < 2\tau\lambda\overline{w_{v,i}}$, $\operatorname{argmin}\mathcal{R}(x_{v,i}) = z_{v,i} - w_{v,i}\operatorname{sgn}(x_{v,i})$; otherwise, $\operatorname{argmin}\mathcal{R}(x_{v,i}) = 0$.

   In fact, if $|z_{v,i}| > w_{v,i}$ and $x_{v,i} \neq 0$, the derivative of $\mathcal{R}_w(x_{v,i})$ is zero (and we have a minimum) for $x_{v,i} = z_{v,i} - w_{v,i}\operatorname{sgn}(x_{v,i})$, that is, $x_{v,i} = z_{v,i} - w_{v,i}$ if $z_{v,i} > w_{v,i}$, and $x_{v,i} = z_{v,i} + w_{v,i}$ if $z_{v,i} < -w_{v,i}$. This is not sufficient: this minimum has to be compared with $\mathcal{R}_w(0)$, which, due to discontinuity, should be lower (see Figure 2.(b)-(c)) This occurs for $(|z_{v,i}| - w_{v,i})^2 < \tau\lambda\overline{w_{v,i}}$, since $\mathcal{R}_w(z_{v,i} - w_{v,i}\operatorname{sgn}(x_{v,i})) = \frac{1}{2\tau}w_{v,i}(2|z_{v,i}| - w_{v,i})$ and $\mathcal{R}_w(0) = \frac{1}{2\tau}z_{v,i}^2$.

We observe that, despite the discontinuity in zero, the case $|z_{v,i}| \le w_{v,i}$ is analogous to soft thresholding. That is, the presence of the $\mathbb{1}(x_v)$ term does not change the position of the minimum (Figure 2.(a)). However, when $|z_{v,i}| > w_{v,i}$ the term $\mathbb{1}(x_v)$ induces to choose zero more often than soft thresholding.

Hence, our procedure to get the minimum of $\mathcal{R}(x_{v,i})$ is given by the mixed soft/hard thresholding. operator $\mathbb{S}_{w,a} : \mathbb{R} \mapsto \mathbb{R}$, defined as follows:

$$\mathbb{S}_{w,a}(x) := \begin{cases} 0 \text{ if } |x| \le w \text{ or } (x-w)^2 \le a \\ x - \operatorname{sgn}(x)w \text{ otherwise.}\end{cases} \tag{12}$$

This is a slight modification of the well-known soft thresholding operator $\mathbb{S}_w : \mathbb{R} \mapsto \mathbb{R}$

$$\mathbb{S}_w(x) := \begin{cases} 0 \text{ if } |x| \le w \\ x - \operatorname{sgn}(x)w \text{ otherwise.}\end{cases} \tag{13}$$

Accordingly, we can write

$$x_{v,i}^+ = \operatorname*{argmin}_{x_{v,i}\in\mathbb{R}}\mathcal{R}(x_{v,i}) = \mathbb{S}_{w_{v,i},\overline{w_{v,i}}}(z_{v,i})$$

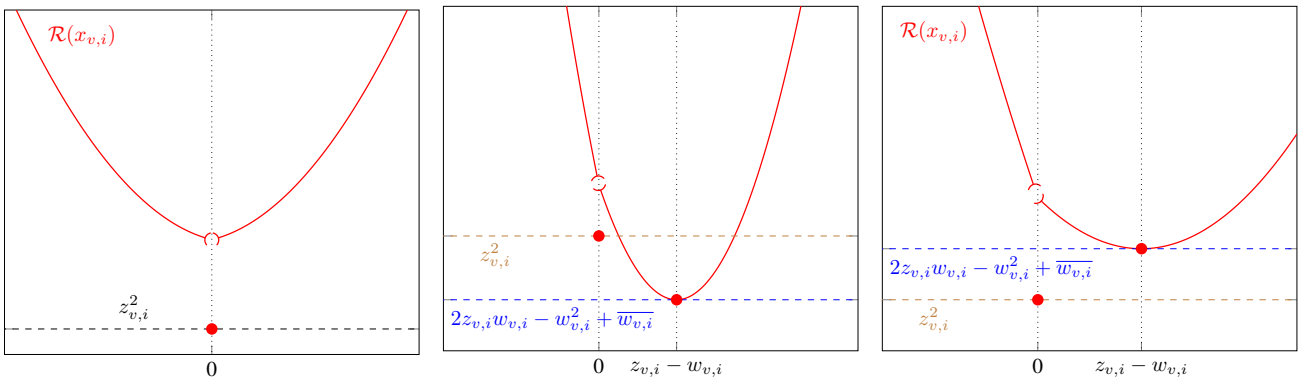
which, if $\frac{1}{\tau} > \|A_v\|_2^2$, implies that ([34, Section 4.1] for details)

$$X = \operatorname*{argmin}_{B\in\mathbb{R}^{n\times V}}\mathcal{R}(X,B). \tag{14}$$

Finally, we conclude that $\mathcal{F}$ decreases:

$$\begin{aligned}\mathcal{F}(X) =& \mathcal{R}(X,X) \ge \mathcal{R}(X^+,X) &(15)\\ &\ge \mathcal{R}(X^+,X^+) = \mathcal{F}(X^+) &(16)\end{aligned}$$

where $X^+ = (x_1^+, \dots, x_V^+)$. The inequality $\mathcal{R}(X,X) \ge \mathcal{R}(X^+,X)$ is guaranteed by LLA [19], [31]. This will be used in next section to prove the convergence.



(a) $|z_{v,i}| < w_{v,i}$ (b) $|z_{v,i}| > w_{v,i}$, $(z_{v,i} - w_{v,i})^2 > \overline{w_{v,i}}$ (c) $|z_{v,i}| > w_{v,i}$, $(z_{v,i} - w_{v,i})^2 < \overline{w_{v,i}}$

Figure 2: $\mathcal{R}(x_{v,i})$ (11) in the cases $|z_{v,i}| < w_{v,i}$ (a) and $|z_{v,i}| > w_{v,i}$ (b)-(c).

The procedure outlined above can be summarized as follows: at each iteration step t, each node v computes $x_{v,i}(t+1) = \mathbb{S}_{w_{v,i}(t),\overline{w_{v,i}}(t)}(z_{v,i}(t))$, for each $i = 1, \dots, n$, where $z_v(t) = x_v(t) + \tau A_v^{\mathsf{T}}\big(y_v - A_v x_v(t)\big)$; after that, if $\mathbb{1}(x_{v,i}(t+1)) \neq \mathbb{1}(x_{v,i}(t))$, then v transmits i to its neighbors.

In conclusion, this procedure solves points 1), 2) and 3) in Section III by using iterative thresholding. However, we observed that the soft/hard shrinkage operator $\mathbb{S}_{w,a}$ (12) tends to oversupply sparsity, which affects the recovery accuracy. To overcome this drawback, we propose to use (13) instead of (12), that is, classical soft thresholding. As this may increase $\mathcal{F}_w(X)$ (specifically, $\mathcal{F}_w(X(t+1)) > \mathcal{F}_w(X(t))$ when $x_{v,i}(t) = 0$, see Figure 3.(c)), we allow the switch from zero to non-zero only for a finite number of times, thus keeping the overall decreasing behavior. In other words, for a finite transient, we perform soft thresholding; after this transient, the zero components are forced to remain zero. In summary, we update $x_{v,i}(t)$ as follows (see Figure 3):

- if $x_{v,i}(t) \neq 0$, we apply soft thresholding: $x_{v,i}(t+1) = \sigma_{w_{v,i}}(z_{v,t})(t)$. This does not guarantee to get the global minimum of $\mathcal{R}(x_{v,i})$, but the global minimum or the second minimum: in both cases, we *always* decrease $\mathcal{R}$;
- if $x_{v,i}(t) = 0$, $x_{v,i}(t+1) = \sigma_{w_{v,i}}(z_{v,t})(t)$ for a finite number of times (during which $\mathcal{R}$ might increase); afterwards, $x_{v,i}(t+1) = 0$.

We remark again that this transient suboptimal modification i) avoids the transmission of real values, ii) improves the performance (see Section VI), and iii) does not affect the convergence properties of the algorithm (see Section V-A).

Bearing all the above in mind, our distributed procedure for the recovery of jointly sparse signals based on IST, DJ-IST in short, is described in Algorithm 1.

It is worth noting that DJ-IST merely requires to transmit information about the support, specifically, the indices of the components that switched from zero to non-zero and vice versa. Since the sensor signals x_v's are in $\mathbb{R}^n$, DJ-IST transmits $\lfloor \log_2 n \rfloor + 1$ bits for each switched component.

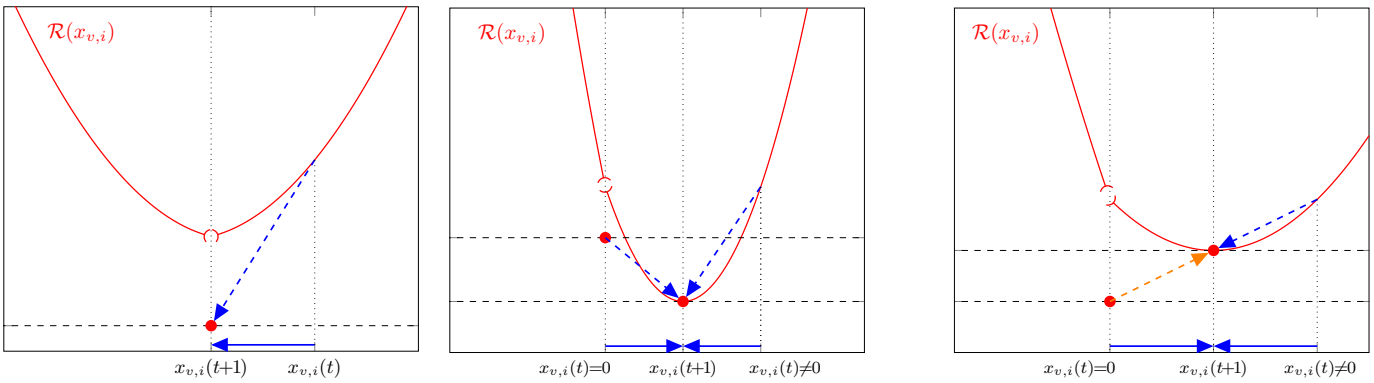
Algorithm 1 DJ-IST

Initialize variables:
For all $v \in \mathcal{V}$, $x_v(0) = A_v^{\mathsf{T}} y_v$; $s_v(0) = [1, 1, \dots, 1]^{\mathsf{T}}$; $p \in \mathbb{N}$ (finite); $\epsilon > 0$, $\tau > 0$, $\lambda > 0$, $\alpha > 0$
$t = 0$
for all $v \in \mathcal{V}$ **do**
  $z_v(t) = x_v(t) + \tau A_v^{\mathsf{T}}\big(y_v - A_v x_v(t)\big)$
  for all $i = 1, \dots, n$ **do**
    Update threshold $w_{v,i}(t) = [\beta - \alpha|x_{v,i}(t)| - \overline{\mathbb{1}(x_{v,i}(t))}]_+$
    Update signal estimate:
    $x_{v,i}(t+1) = \mathbb{S}_{\lambda\alpha w_{v,i}(t)}(z_{v,i}(t))$
    if $x_{v,i}(t) = 0$ and $c_{v,i}(t) \geq p$ **then**
      $x_{v,i}(t+1) = 0$
    end if
    if $x_{v,i}(t) = 0$ and $x_{v,i}(t) \neq 0$ **then**
      $c_{v,i}(t+1) = c_{v,i}(t) + 1$
    end if
    if $\mathbb{1}(x_{v,i}(t+1)) \neq \mathbb{1}(x_{v,i}(t))$ **then**
      Transmit index i to the neighbors
    end if
  end for
  if $\|x_v(t+1) - x_v(t)\|_2 < \epsilon$ **then**
    Node v stops
  else
    $t \longleftarrow t + 1$
  end if
end for

A. Other iterative algorithms for Lasso

At the beginning of the section, the use of iterative thresholding was naturally motivated by its adaptability to decrease the non-convex functional $\mathcal{F}$ in (6), which presents ℓ_1 and ℓ_0 penalization terms.

In the literature, methods faster than iterative thresholding have been proposed to solve convex problems as Lasso. For example, the alternating direction method of multipliers (ADMM, [38], [39]), and the fast iterative thresholding algorithm (FISTA, [44]) have been shown to be very efficient. In



(a) $|z_{v,i}(t)| < w_{v,i}(t)$

(b) $|z_{v,i}(t)| > w_{v,i}(t)$, $(z_{v,i}(t) - w_{v,i}(t))^2 > \overline{w}_{v,i}(t)$

(c) $|z_{v,i}(t)| > w_{v,i}(t)$, $(z_{v,i}(t) - w_{v,i}(t))^2 < \overline{w}_{v,i}(t)$

Figure 3: Dynamics of $\mathcal{R}(x_{v,i}(t))$ when $x_{v,i}(t{+}1) = \mathbb{S}_{w_{v,i}}(x_{v,i}(t))$. The arrows depict the movements of $x_{v,i}(t)$ and $\mathcal{R}(x_{v,i}(t))$. In the case (c), if $x_{v,i}(t) = 0$, $\mathcal{R}(x_{v,i}(t)) < \mathcal{R}(x_{v,i}(t+1))$ (orange arrow). This increasing movement is allowed only for a finite number of times, after which if $x_{v,i}(t) = 0$, we fix $x_{v,i}(t+1) = 0$. In this way, the definitive behavior of $\mathcal{R}(x_{v,i}(t))$ is non-increasing.

principle, such methods cannot be applied to (6) due to the non-convexity of $\mathcal{F}$. Through this section, however, we have reduced the step that updates $X(t)$ to an IST step (with forced stabilization of the null components after a finite transient), which means that we simply decrease the Lasso part of $\mathcal{F}$, and the role of ℓ_0 is only to stop the switches from zero to non-zero.

From this perspective, we could consider again methods as ADMM and FISTA to update $X(t)$. However, we observed that such methods are somehow too fast for our problem. In fact, if the procedure is too fast, nodes tend to estimate their signals support based on their local measurements, *i.e.*, without taking into account other nodes information. This causes some transient instability in which many support switches occur, which implies many more transmissions, thereby penalizing the communication cost. In conclusion more conservative methods ultimately reduce the number of transmissions, which makes the slow IST more efficient. In order to illustrate these observations, in Section VI-F we will show some numerical simulations based on ADMM.

V. Convergence of DJ-IST

In this section, we prove that DJ-IST converges. We first show the the numerical convergence and the support stabilization, and we then exploit them to prove the point convergence.

A. Numerical convergence

We now prove the numerical convergence (or asymptotic regularity) of the sequence $X(t)$ produced step by step by Algorithm 1, namely

$$\lim_{t\to+\infty} \|X(t+1) - X(t)\|_F^2 = 0$$

We remark that for the convergence analysis we do not take into consideration the fact that for a finite number of steps, increases of $\mathcal{R}$ are allowed, as they clearly have no effect on the asymptotic properties of the algorithm. From now on, we then consider $t \geq t_0$, where t_0 is any fixed time step after the finite transient.

Proposition 1. *Given the sequence $X(t)$ generated by DJ-IST (Algorithm 1), $\{\mathcal{F}(X(t))\}_{t\in\mathbb{N}}$ for $t \geq t_0$ is non-increasing, and admits the limit. Moreover, if $\tau < \|A_v\|_2^{-2}$ for all $v \in \mathcal{V}$, $X(t)$ is numerically convergent.*

Proof. By (15) and following discussion, for any for $t \geq t_0$, $\mathcal{F}(X(t)) \geq \mathcal{F}(X(t+1))$, that is, $\mathcal{F}(X(t))$ is non-increasing. As it is lower bounded ($\mathcal{F}(X) \geq 0$ for any $X \in \mathbb{R}^{n\times V}$), then it admits the limit. Hence, $\mathcal{F}(X(t)) - \mathcal{F}(X(t+1)) \to 0$. On the other hand,

$$\begin{aligned}
&\mathcal{F}(X(t)) - \mathcal{F}(X(t+1)) \\
&= \mathcal{R}(X(t), X(t)) - \mathcal{R}(X(t+1), X(t+1)) \\
&\geq \mathcal{R}(X(t+1), X(t)) - \mathcal{R}(X(t+1), X(t+1)) \\
&\geq \sum_{v\in\mathcal{V}} (x_v(t+1) - x_v(t))^\mathsf{T} (I - \tau A_v^\mathsf{T} A_v)(x_v(t+1) - x_v(t)) \\
&\geq 0.
\end{aligned}$$

The last inequality is due to the positive definiteness of $I - \tau A_v^\mathsf{T} A_v$ guaranteed by the hypothesis $\tau < \|A_v\|_2^{-2}$. We thus conclude that $\|x_v(t+1) - x_v(t)\|_2^2 \to 0$ for any $v \in \mathcal{V}$ and that $\lim_{t\to+\infty} \|X(t+1) - X(t)\|_F^2 = 0$.

□

Furthermore, we can easily observe that support stabilizes at a finite time.

Theorem 1. *There exists a time $t_1 \in \mathbb{N}$ at which the sequence $\mathbb{1}(X(t))$ stabilizes, that is, $\mathbb{1}(X(t))$ is constant for any $t \geq t_1$.*

Proof. After a finite number of allowed switches, no more switches from zero to non-zero are possible for DJ-IST, say $x_{v,i}(t) = 0$, then $x_{v,i}(t+1) = 0$, for any $v \in \mathcal{V}$, $1 = 1, \ldots, n$. This is sufficient to state that the support stabilizes. In particular, we call t_1 the time at which all the components of all the nodes have stabilized their status. □

Alternatively, this result could be easily deduced from Proposition 1. Since $X(t)$ numerically converge and the support stabilize, we notice that also $W(t)$ numerically converge.

B. Point convergence

We now leverage numerical convergence and support stabilization to prove rigorous point convergence.

Once the support estimation has stabilized, our main goal should be considered achieved. No more communication is necessary and the signal estimate (say, the estimate of the non-zero values) could be performed by each node singularly by a least squares method, as done in [18].

However, with DJ-IST it is not necessary to split the recovery into two different procedures, one for the estimate of the support and one for the estimate of the non-zero values. Notice that splitting the recovery into two different procedures is more critical when k is not known, as there is no secure criterion to establish when the support has stabilized. We now show that one can run DJ-IST also after the support stabilization and get the convergence of $X(t)$. In the previous section, we have already proved the numerical convergence, which provides a practical stopping criterion: at any $t \in \mathbb{N}$, each node should store $x_v(t)$ and $x_v(t-1)$ and stop when the distance between the two iterates is below a fixed threshold depending on the machine epsilon. In this section, we propose a rigorous point convergence proof and give a description of the convergence points.

Let us consider the system evolution after support stabilization. First of all, we notice that the problem is no more distributed: communications actually stop and each node $v \in \mathcal{V}$ proceeds individually.

As the zeros are now fixed, let us now describe the evolution of the non-zero components of each v. Let us call $\widehat{\Omega}_v \subset \{1, \dots, n\}$ the active set, *i.e.* the estimated support for node v, which is constant after support stabilization. We define the partition: $\widehat{\Omega}_v = \widehat{\Omega}_{v,1}(t) \cup \widehat{\Omega}_{v,2}(t)$ where

$$\widehat{\Omega}_{v,1}(t) := \{i \in \{1, \dots, n\} \text{ s.t. } w_{v,i}(t) > 0\}$$

and $\widehat{\Omega}_{v,2} := \widehat{\Omega}_v \setminus \widehat{\Omega}_{v,1}$, that is,

$$\widehat{\Omega}_{v,2}(t) = \{i \in \{1, \dots, n\} \text{ s.t. } w_{v,i}(t) = 0\}.$$

First, we remark that the signs of the non-zero components are definitely constant. To see this, suppose the sign changes in the next iteration, *e.g.* $x_{v,i}(t) > 0$ and $x_{v,i}(t+1) < 0$. Given the numerical convergence, large deviations between consecutive iterations are not possible, and thus we expect $x_{v,i}(t) \in \widehat{\Omega}_{v,1}(t)$, so that $x_{v,i}(t) < \frac{\beta - \widehat{\mathbb{1}}(x_{v,i}(t))}{\alpha}$. We have then $w_{v,i}(t) > 0$, and in particular the more $x_{v,i}(t)$ is close to zero, the more $w_{v,i}(t)$ is large, then we can consider $w_{v,i}(t) \geq \epsilon > 0$. To switch the sign we must have $z_{v,i}(t) > \alpha w_{v,i}(t)$ and $z_{v,i}(t+1) < -\alpha w_{v,i}(t)$; however, this is not possible as $z_{v,i}(t)$ numerically converges as well, and after a finite time it cannot overstep an interval of length $2\alpha w_{v,i}(t) > 2\epsilon > 0$. Following this rationale, an intermediate step in which $|z_{v,i}(t)| < \alpha w_{v,i}(t)$ is expected, which entangles $x_{v,i}(t)$ into zero.

Bearing this in mind, the evolution of the non-zero components can be expressed as follows. Let $A_{\widehat{\Omega}_v}$ be A_v limited to the columns that belong to $\widehat{\Omega}_v$. We have

$$\begin{aligned} \Gamma_v :& \mathbb{R}^{\widehat{k}_v} \mapsto \mathbb{R}^{\widehat{k}_v} \\ \Gamma_v(x) &= M_v(x)x + c_v(x) \end{aligned} \tag{17}$$

where

$$M_v(x) \in \mathbb{R}^{\widehat{k}_v \times \widehat{k}_v}, \quad M_v(x) = \alpha^2 D_v(x) + I_v - \tau A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v}$$
$$c_v(x) \in \mathbb{R}^{\widehat{k}_v}, \quad c_v(x) = -D_v(x)\alpha \mathrm{sgn}(x)(\beta - \widehat{\mathbb{1}}_v) + \tau A_{\widehat{\Omega}_v}^{\mathsf{T}} y_v$$

and I_v is the identity matrix of dimensions $\widehat{k}_v \times \widehat{k}_v$; $D_v(x)$ is the binary diagonal matrix which has a 1 in position (i,i) if $x_{v,i} \in \widehat{\Omega}_{v,1}$, and zero otherwise; $\widehat{\mathbb{1}}_{v,i} = \overline{\mathbb{1}(x_{v,i}(t_1))}$, where t_1 is the support stabilization time, then $\widehat{\mathbb{1}}_v$ is constant.

$M_v(x)$ is positive definite for any $x \in \mathbb{R}^{\widehat{k}_v}$, and whenever a component of x_v is in $\widehat{\Omega}_{v,1}$, the transition matrix $M_v(x)$ is expansive if $A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v}$ has not maximum rank. Iterating $\Gamma_v(x) = M_v(x)x + c_v(x)$ we then expect that all the components of x_v will blow up at infinity, but actually this is not the case because when $|x_{v,i}| > \frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$, we move to regime $\widehat{\Omega}_{v,2}$, in which the system turns out to be a simple gradient descent that converges to a minimum of $\|A_{\widehat{\Omega}_v} x - y_v\|$. This proves the following Lemma.

Lemma 1. *For any $v \in \mathcal{V}$, $t \in \mathbb{N}$, $x_v(t)$ is bounded.*

The dynamical system of (17) is a switched linear system: when $x_{v,i}(t)$ switches from $\widehat{\Omega}_{v,1}$ to $\widehat{\Omega}_{v,2}$, the entry (i,i) of D_v switches from 1 to 0, and vice versa. Possible oscillations between the two regions make the convergence proof more complicated and technical. To simplify it, we do the following realistic assumption.

Assumption 2. *For any $v \in \mathcal{V}$ and $t \in \mathbb{N}$, $\max |x_{v,i}(t)| < \frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$, that is, $x_{v,i}(t) \in \widehat{\Omega}_{v,1}$.*

This assumption is commonly fulfilled as generally we set α much smaller than β. Therefore, $|x_{v,i}(t)| \geq \frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$ implies $\mathcal{F}(X)(t)$ of the order of $\frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$, which is very high. For example, in our simulations (Section VI), we set $\alpha = 5 \cdot 10^{-4}$ and $\beta = 1.1$, which implies $\mathcal{F}(X(t))$ of order 10^6 for $|x_{v,i}(t)| \geq \frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$. Therefore, it suffices to set a reasonable initial condition to have $\mathcal{F}(X(0))$ smaller than such values: since $\mathcal{F}(X(t))$ is not increasing, this guarantees that $|x_{v,i}(t)$ will never exceed $\frac{\beta - \widehat{\mathbb{1}}_{v,i}}{\alpha}$.

Under Assumption 2, the evolution of our system is simply linear:

$$\begin{aligned} \Gamma_v :& \mathbb{R}^{\widehat{k}_v} \mapsto \mathbb{R}^{\widehat{k}_v} \\ \Gamma_v(x) &= M_v x + c_v \end{aligned} \tag{18}$$

where

$$M_v = (\alpha^2 + 1)I_v - \tau A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v} \in \mathbb{R}^{\widehat{k}_v \times \widehat{k}_v}$$
$$c_v = \alpha \mathrm{sgn}(x)(\beta - \widehat{\mathbb{1}}_v) + \tau A_{\widehat{\Omega}_v}^{\mathsf{T}} y_v \in \mathbb{R}^{\widehat{k}_v}$$

From previous observations, we know that $x_{v,i}(t)$ is bounded, so such M_v cannot be expansive. We therefore

conclude that $A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v}$ must have maximum rank. Assuming that the components of A_v are randomly chosen according a continuous distribution, $A_v^{\mathsf{T}} A_v$ has rank m; since $A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v}$ has dimension $\widehat{k}_v$, we conclude that it can have maximum rank $\widehat{k}_v$ only if $\widehat{k}_v \leq m$. We observe that this makes sense, as this is the case for iterative soft thresholding [45], which is the basis for our algorithm. This condition is necessary but also sufficient to have maximum rank, provided that $\tau A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v}$ has no eigenvalues equal to α^2 (if A_v is random, this occurs with probability 0). Moreover, if α is sufficiently small, we have $\|M_v\|_2 < 1$

Finally, we have the following convergence theorem.

Theorem 2. *For a sufficiently small α, the sequence $X(t)$ generated by DJ-IST (Algorithm 1) converges to a local minimum of $\mathcal{F}(X)$. Moreover, for each $v \in \mathcal{V}$, the non-zero components of $x_v(t)$ converge to*

$$[I_v - M_v]^{-1} c_v = [\tau A_{\widehat{\Omega}_v}^{\mathsf{T}} A_{\widehat{\Omega}_v} - \alpha^2 I_v]^{-1} [\alpha s_v (\beta - \widehat{\mathbb{1}}_v) + \tau A_{\widehat{\Omega}_v}^{\mathsf{T}} y_v] \tag{19}$$

where $s_v = sgn(x_v(t_1))$, t_1 being the support stabilization time.

Proof. For a sufficiently small α, $\|M_v\|_2 < 1$, that is, the map (18) is contractive. Therefore, a fixed point exists and convergence to it is guaranteed (no matter which is the initial point) by the Banach fixed-point theorem. In particular, iterating the map (18) we obtain a geometric series that converges to (19).

This concludes the convergence of the non-zero components, which along with support stabilization proved in Theorem 1 gives the convergence.

We remark that the point (19) turns out to be the unique minimum of

$$\tau \left\| A_{\widehat{\Omega}_v} x - y_v \right\| + \sum_{i=1}^{n} 2\beta\alpha |x_i| - \frac{1}{2}\alpha^2 x_i^2$$

and, as a consequence, a local minimum of $\mathcal{F}(X)$. In fact, if we perturb the non null components we increase $\mathcal{F}$ due to the last statement, while if we perturb the zero components, the indicator function switch to 1 and cause a sure increase of $\mathcal{F}$. □

Regarding the convergence point (19), we observe that this coincides with the true value if $x_v^\star = \frac{1}{\alpha} s_v (\beta - \widehat{\mathbb{1}}_v)$, otherwise a bias is present. This was expected as ℓ_1 minimum is known to be bias proportionally to the the ℓ_1 weight. In our reweighted ℓ_1 setting, however an accurate choice of β and α could reduce this bias. Such optimization will be focus of our future work.

VI. Numerical results

In this section, we show the results of some numerical simulations and compare the performance of DJ-IST with the state-of-the-art algorithms DC-OMP 1 and DC-OMP 2 [18].

A. DC-OMP 1 and DC-OMP 2

The rationale behind DC-OMP 1 [18, Algorithm 3] is the following: each node performs a step of OMP and computes an index candidate (by evaluating the largest correlation between residual and columns of the sensing matrix) to add to the support; the candidates are then locally shared, and the candidates with more than one occurrence are added to the support, except for the case that those candidates do not change the support (in this case, each node introduces its own candidate); if all the candidates have one occurrence, each node adds its own candidate. A slight modification is considered when the communication is complete. Notice that DC-OMP 1 is very similar to DiOMP [16], with some differences in the voting procedure, which makes DC-OMP 1 more reliable. In DC-OMP 2 [18, Algorithm 4], instead, each node locally shares not only the index candidate, but all the correlations between residual and columns of its sensing matrix. The index candidate is then chosen fusing the correlations and then transmitted to all the network via multi-hop communication. In DC-OMP 2 more information is shared with respect to DC-OMP 1, then better performance can be expected.

The goal of this section is to numerically prove that DJ-IST is a good trade-off between DC-OMP 1 and DC - OMP 2, in terms of support reconstruction accuracy and use of the communication links.

B. Simulations setting

For all our experiments, the original signals $x_v^\star$ have joint support generated uniformly at random, and the non-zero elements are drawn from a standard Gaussian distribution. The entries of the sensing matrices are generated according to a standard Gaussian distribution as well, and then normalized by $\sqrt{m}$. Results are averaged over 250 different runs, obtained by generating 50 different sets of $x_v^\star$ and trying 5 different sensing matrices for each. We stop the algorithm at time $T = \min\{t \in \mathbb{N} \text{ s.t. } |x_{v,i}(t+1) - x_{v,i}(t))| < \epsilon = 10^{-5}$, for all $v \in \mathcal{V}, i = 1, \ldots, n\}$. The parameters λ, α, β and τ have been empirically set; in all our simulations, $\lambda = 1$, $\alpha = 5 \times 10^{-4}$, $\beta = 1.1$, $\tau = 2e - 2$. The parameter p is not actually fixed, as naturally few switches from zero to non-zero occur (in all our simulations, we observed at most 9 switches).

C. Support recovery performance

We evaluate two performance metrics for the support: the average support error (ASE), defined as

$$\text{ASE} = \sum_{v \in \mathcal{V}} \frac{\|\mathbb{1}(x_v^\star) - \widehat{\omega}_v\|_0}{nV} \tag{20}$$

and the probability of exact support recovery (PESR)

$$\text{PESR} = \sum_{v \in \mathcal{V}} \frac{\mathbb{I}(\mathbb{1}(x_v^\star) - \widehat{\omega}_v)}{V} \tag{21}$$

where $\mathbb{I}(x)$ is the function from $\mathbb{R}^n$ to $\mathbb{R}$ that returns 1 when the vector $x = (0, 0, \ldots, 0)^{\mathsf{T}} \in \mathbb{R}^n$ and 0 otherwise. PESR assesses how many sensors estimate the right support, while ASE measures how large is the error in the support for each sensor, on average.

In Figure 4, we show the ASE and the PESR for a network of $V = 10$ nodes, varying of the number of measurements per node m between 4 and 32. We show both the complete

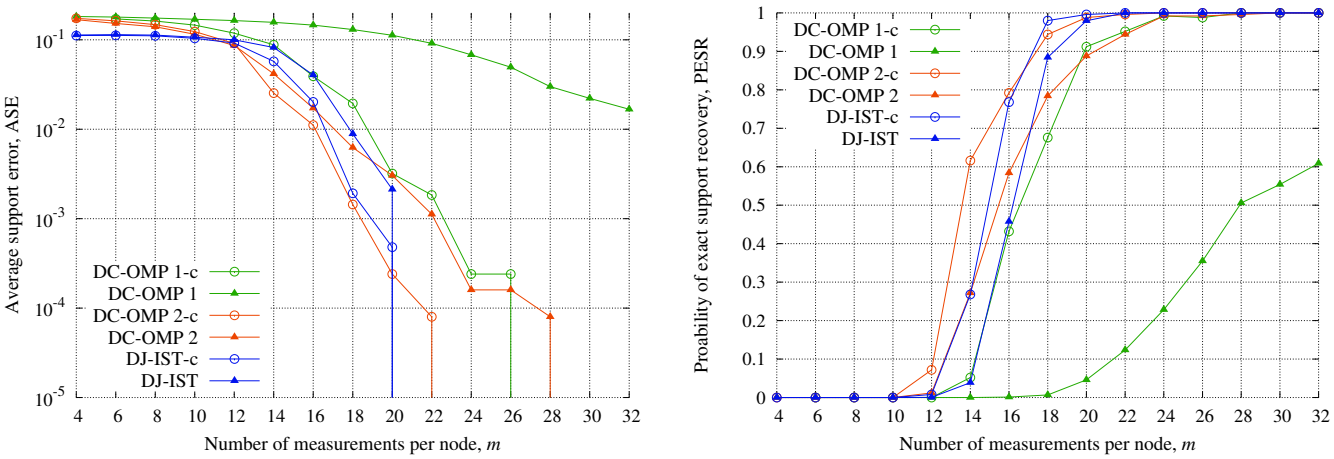


Figure 4: ASE (left) and PESR (right) as a function of m, $V = 10$, $\lambda = 1$, $\alpha = 5 \times 10^{-4}$, $\beta = 1.1$, $\tau = 2 \times 10^{-2}$.

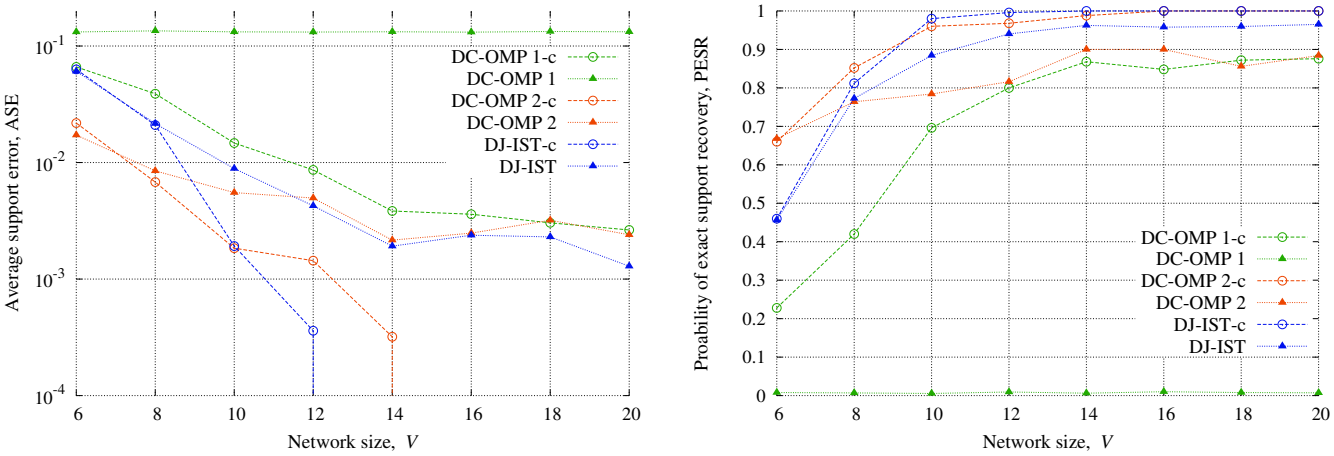


Figure 5: ASE (left) and PESR (right) as a function of V, $m = 18$, $\lambda = 1$, $\alpha = 5 \times 10^{-4}$, $\beta = 1.1$; $\tau = 8 \times 10^{-3}$ for complete graphs, except for $V \in 6, 8$ where $\tau = 3 \times 10^{-3}$; $\tau = 2 \times 10^{-2}$ for 5-regular graphs, except for $V \in 6, 8$ where $\tau = 8 \times 10^{-3}$

graph case (indicated by the postfix '-c') and the regular case with $d = 5$ (say, each node has 4 neighbors). The ASE is shown in logarithmic scale: a vertical line indicates the m beyond which the ASE is exactly zero. We immediately notice that DJ-IST (in both complete and non complete regimes) achieves null ASE with a smaller m than all the other methods. Specifically, we observed that $m = 22$ is sufficient for DJ-IST to have perfect support detection, while $m = 24, 28, 30$ are necessary respectively for DC-OMP 2-c, DC-OMP 1-c, DC-OMP 2. We further remark that DC-OMP 1 never gets zero in the considered range.

We also notice that for any considered m DJ-IST performs better than DC-OMP 1 and less worse than DC-OMP 2 (except for vary small m, where DJ-IST is the best). Recalling that DC-OMP 2 always envisages a complete topology (as it exploits global (multihop) communication in the non-complete case), the fact that DJ-IST-c is very close to DC-OMP 2 is remarkable. Analogous considerations can be done for the PESR curve.

In Figure 5, we show the ASE and the PESR for fixed $m = 18$ and varying V. Again, we appreciate that DJ-IST outperforms DC-OMP 1, while the PESR of DJ-IST is better than that of DC-OMP 2 in the non-complete regime, for large networks.

We remark that for non-complete topologies, support agreement among the nodes is not guaranteed; analytical conditions to get consensus will be subject of future research. However, if necessary, a consensus algorithm can be run after our procedure to obtain the same support over all the network.

D. *Signal estimation performance*

In addition to the support recovery analysis, we report some observations about the signal estimation accuracy of DJ-IST. In fact, as already remarked, DJ-IST, as a difference from [16], [18], performs both support and signal estimation.

In Figure 6 we depict the mean relative square error (RSE) which we define as

$$\text{RSE} = \frac{\sum_{v \in \mathcal{V}} \|x_v^\star - \widehat{x}_v\|_2^2}{\sum_{v \in \mathcal{V}} \|x_v^\star\|_2^2}. \tag{22}$$

The used parameters are the ones used in the experiments presented in the previous paragraph, and RSE and ASE are shown as functions of m (left) and V (right). As we are adopting a logarithmic scale, we visualize a vertical line when the ASE goes to zero. In these graphs, we can appreciate

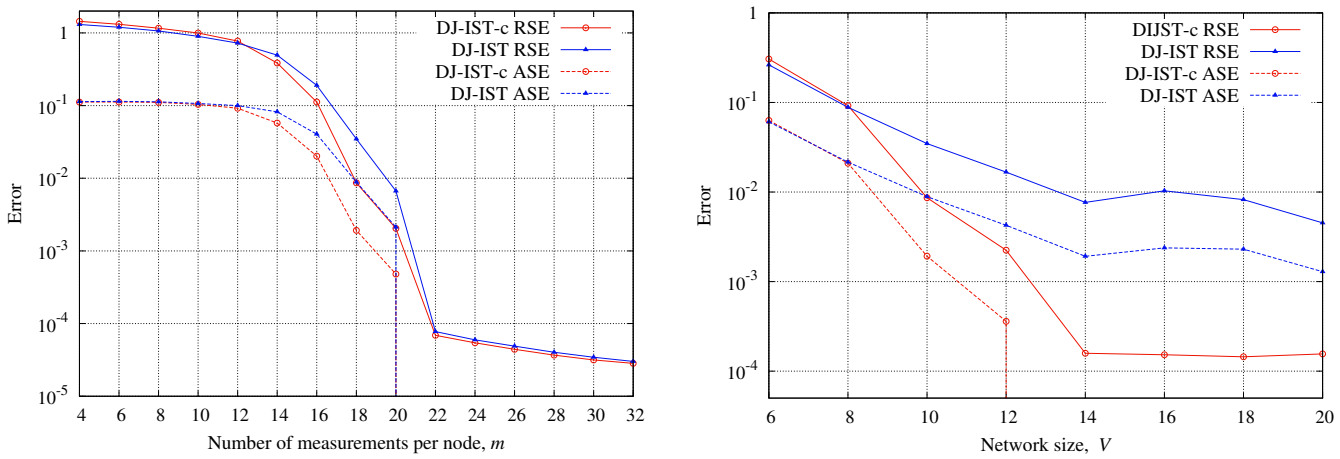


Figure 6: RSE and ASE as a function of m (left) and V (right).

that the RSE follows the behavior of the ASE. A small bias occurs in the RSE when the ASE is null, which is expected due to our Lasso approach. The reweighting method reduces the Lasso bias, but does not totally remove it, even though because of Assumption 2: shrinkage is reduced, but actually never removed for non-zero coefficients.

E. Analysis of transmission efficiency

We now analyze the transmission efficiency of DJ-IST, compared to DC-OMP 1 and 2 [18], in terms of number of transmitted bits over each network link. The range of transmitted bits can be analytically evaluated for all the three algorithms, as we now show. Afterwards, we will present some statistics from numerical simulations.

Let us consider the non-complete graph case, and for simplicity let us assume a d-regular topology. In DJ-IST and DC-OMP 1 only indices in $\{1,\dots,n\}$ are transmitted, then each index can be encoded with $\lfloor \log_2 n \rfloor + 1$ bits. In DC-OMP 1, each node $v \in V$ transmits to its $d-1$ neighbors its candidate for activation, namely, the coefficient it would add to the support; afterwards, basically each coefficient with more than two votes is added to the support. Therefore, at each step a maximum a $\lfloor \frac{d}{2} \rfloor$ coefficient could be added, and to complete the support the minimum possible number of step is $\lceil k/\lfloor \frac{d}{2} \rfloor \rceil$, while the maximum is k (one coefficient at each step; we recall that k has to be exactly known in DC-OMP approach, which is not required for DJ-IST). In conclusion, in DJ-IST the total number of bits transmitted over a link is in the range $V(d-1)(\lfloor \log_2 n \rfloor + 1)\left[\lceil k/\lfloor \frac{d}{2} \rfloor \rceil, k\right]$.

In DC-OMP 2, the nodes share with neighbors the correlation vector in $\mathbb{R}^n$; assuming q bits for each real value, this amounts to $V(d-1)qn$ bits per iteration. The nodes use such information to choose their own candidate coefficient, and they broadcast it to all the network, which amounts to $V(V-1)(\lfloor \log_2 n \rfloor + 1)$. The voting procedure to build the support is analogous to DC-OMP 1. Hence, the total number of transmitted bits is in the range $V\left[(d-1)q + (V-1)(\lfloor \log_2 n \rfloor + 1)\right]\left[\lceil k/\lfloor \frac{d}{2} \rfloor \rceil, k\right]$.

Differently from DC-OMP strategies, in DJ-IST all the coefficients start as active and then, hopefully, $n-k$ of them are switched to zero. Each $v \in V$ communicates to neighbors the switches for non-zero to zero, and vice versa. If all the nodes remain non-zero, no communications occurs, while the maximum is $2pnV(d-1)$, where p is the maximum number of switches from zero to non-zero discussed in Section IV[2].

We sum up these ranges in Table I. Next, in Tables II and III, we show transmission load statistics taken from our simulations over regular graphs with degree $d = 5$ (250 runs). Real values are assumed to be quantized over $q = 16$ bits.

Table I: Transmitted bits: ranges for d-regular topologies ($r = \lfloor \log_2 n \rfloor + 1$)

Algorithms	Min	Max
DC-OMP 1	$V(d-1)r\lceil k/\lfloor \frac{d}{2} \rfloor \rceil$	$V(d-1)rk$
DC-OMP 2	$V\left[(d-1)q + (V-1)r\right]\lceil k/\lfloor \frac{d}{2} \rfloor \rceil$	$V\left[(d-1)q + (V-1)r\right]k$
DJ-IST	0	$2pnV(d-1)$

Table II: Transmitted bits: statistics over all the simulations with $n = 100$, $k = 10$, $V = 10$, $m \in \{4, 6, 8, \dots, 32\}$)

Algorithms	Min	Max	Mean
DC-OMP 1	2520	2800	2795
DC-OMP 2	193890	387780	298590
DJ-IST	29288	39508	32938

[2] $2pn$ stands for the worst case in which all the coefficients oscillate as long as can, and then switch off to zero.

Table III: Transmitted bits: statistics over all the simulations with $n = 100$, $k = 10$, $V \in \{6, 20\}$, $m = 18$)

Algorithms $V = 6$	Min	Max	Mean
DC-OMP 1	1512	1680	1673
DC-OMP 2	193050	386100	328957
DJ-IST	16828	34552	21750
Algorithms $V = 20$	**Min**	**Max**	**Mean**
DC-OMP 1	4480	5600	5570
DC-OMP 2	261320	522640	373687
DJ-IST	61236	92624	69568

F. DJ-ADMM

In Section IV-A, we intuitively explained that replacing the IST step in DJ-IST (Step 7 in Algorithm 1) with faster Lasso decreasing algorithms is not expected to improve the performance. We now show an example: we replace IST with ADMM [38], The settings are as follows: $\lambda = 1$, $\alpha = 5{\times}10^{-3}$. For each $v \in \mathcal{V}$, we consider the augmented Lagrangian

$$\begin{aligned}\mathcal{L}(x_v, z_v; \mu_v) = \frac{1}{2}\left\|y_v - A_v x_v\right\|_2^2 + \lambda\alpha \sum_{i=1}^{n} w_{v,i}|z_{v,i}| \\ + \rho\mu_v^{\mathsf{T}}(x_v - z_v) + \rho\left\|x_v - z_v\right\|_2^2\end{aligned} \tag{23}$$

where $\rho > 0$ (here we fix $\rho = 1$), x_v, z_v, $\mu_v \in \mathbb{R}^n$. Given $\mathrm{d}(\mu_v) = \min_{x_v, z_v} \mathcal{L}(x_v, z_v; \mu_v)$, at each step, ADMM decreases the functional $\mathcal{L}(x_v, z_v; \mu_v) - 2\mathrm{d}(\mu_v)$ [46, Theorem 3.1]. Specifically the ADMM step for Lasso is as follows (see [38, Section 6.4]):

$$\begin{aligned}x_v(t+1) &= \operatorname*{argmin}_{x_v} \mathcal{L}(x_v, z_v(t); \mu_v(t)) \\ &= (A_v^{\mathsf{T}} A_v + \rho I)^{-1}[A_v^{\mathsf{T}} y_v + \rho(z_v(t) - \mu_v(t))] \\ z_v(t+1) &= \operatorname*{argmin}_{z_v} \mathcal{L}(x_v(t), z_v; \mu_v(t)) \\ &= \mathbb{S}_{\lambda\alpha w_v(t)/\rho}[x_v(t+1) + \mu_v(t)]. \\ \mu_v(t+1) &= \mu_v(t) + x_v(t+1) - z_v(t+1).\end{aligned}$$

We name DJ-ADMM the algorithm that we obtain by replacing IST with ADMM in DJ-IST, with the usual forced stopping of the null components above a switch threshold p. In our simulations, we observed that no more than 5 switches from zero to non-zero occurred using DJ-ADMM, and as for DJ-IST, in the practice we did not set p in advance.

In Figure 7 we compare DJ-IST and DJ-ADMM for varying m, averaged over 250 runs. The setting is the one described in Section VI-B, with regular topology with degree 5. First, we show that the support reconstruction accuracy, evaluated in terms of ASE, is very similar. When the support is exactly recovered, the RSE of DJ-ADMM achieves 10^{-6}, while DJ-IST is around 10^{-5}, due to the bias that can be evaluated from (19).

We further observe that DJ-ADMM is much faster in terms of number of iterations (second graph of Figure 7), but requires a larger number of bit transmissions (third graph). As already explained, this is expected as ADMM forces a faster decrease of the Lasso, which may produce conflicts with the information gathered from the network; the behavior of the single node is then too aggressive, which causes more switches, hence more transmissions, if compared to DJ-IST. However, the number of transmissions of DJ-ADMM is still of the order of DJ-IST. This makes DJ-ADMM suitable for those cases in which velocity is desired.

Regarding the number of transmitted bits, we remark the peak (for both DJ-IST and DJ-ADMM) for mid values of m. The reason is that when few measurements are available, each node has less information to communicate; on the other hand, many measurements allow a faster convergence and less transmissions. Thus, it is in the intermediate case that the network has its most intense activity.

VII. Conclusion

In this paper, we have proposed DJ-IST, a distributed soft thresholding algorithm to recover jointly sparse signals. The shrinkage thresholds are reweighted at each step, based on information on the support coming from the network. DJ-IST estimates both the support and the non-zero values of the unknown signals. DJ-IST is proved to converge to a minimum of a suitable cost functional with concave penalization. Interestingly, DJ-IST can be interpreted as a distributed reweighted ℓ_1 minimization algorithm. In terms of support recovery accuracy, DC-OMP 2 is the state-of-the-art method. Numerical simulations show that DJ-IST has a performance close to DC-OMP 2, but significantly outperforms it in terms of transmission efficiency (namely, number of transmitted bits per link). On the other hand, DC-OMP 1 is the state-of-the-art method in terms of transmission efficiency, but its performance is shown to be worse than DJ-IST. In conclusion, DJ-IST is an optimal trade-off between recovery performance and energy saving capability, which makes it more suitable than greedy procedures.

The scheme of DJ-IST seems to be applicable to other jointly sparse models, like JSM-1 and JSM-3 [8], that have been recently tackled with distributed algorithms [47], [48]. Moreover, we remark that DJ-IST could be used in case of recovery of a unique common signal [49] to improve the transmission efficiency [50], [51]: sharing information about the support instead of transmitting the whole signal's estimate may dramatically reduce the communication load. These points will be subject of our future work.

References

[1] J. A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1847–1862, 2010.

[2] F. Zeng, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multihop cognitive networks," *IEEE J. Sel. Top. Sign. Proces.*, vol. 5, no. 1, pp. 37–48, 2011.

[3] N. Yu, T. Qiu, F. Bi, and A. Wang, "Image features extraction and fusion based on joint sparse representation," *IEEE J. Sel. Top. Sign. Proces.*, vol. 5, no. 5, pp. 1074–1082, Sept 2011.

[4] X.-T. Yuan, X. Liu, and S. Yan, "Visual classification with multitask joint sparse representation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4349–4360, Oct 2012.

[5] W. Li, Y. Zhou, N. Poh, F. Zhou, and Q. Liao, "Feature denoising using joint sparse representation for in-car speech recognition," *IEEE Sig. Proc. Letters*, vol. 20, no. 7, pp. 681–684, July 2013.

[6] S. Shekhar, V. Patel, N. Nasrabadi, and R. Chellappa, "Joint sparse representation for robust multimodal biometrics recognition," *IEEE Trans. Patt. Ana. & Mach. Intel.*, vol. 36, no. 1, pp. 113–126, 2014.

[7] D. Baron, M. F. Duarte, M. B. Wakin, S. Sarvotham, and R. G. Baraniuk, "Distributed compressive sensing of jointly sparse signals," in *Asilomar Conf. Signals, Sys., Comput.*, 2005, pp. 1537–1541.

[8] M. F. Duarte, S. Sarvotham, D. Baron, M. B. Wakin, and R. G. Baraniuk, "Distributed compressed sensing of jointly sparse signals," in *Proc. 39th Asilomar Conf. Signals, Systems and Computers, Pacific Grove, CA*, 2005.

[9] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, pp. 1289 – 1306, 2006.

[10] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *IEEE Trans. Signal Process.*, vol. 54, no. 12, pp. 4634–4643, Dec 2006.

[11] M. Davies and Y. Eldar, "Rank awareness in joint sparse recovery," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 1135–1146, Feb 2012.

[12] A. Kyrillidis and V. Cevher, "Combinatorial selection and least absolute shrinkage via the clash algorithm," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2012, pp. 2216–2220.

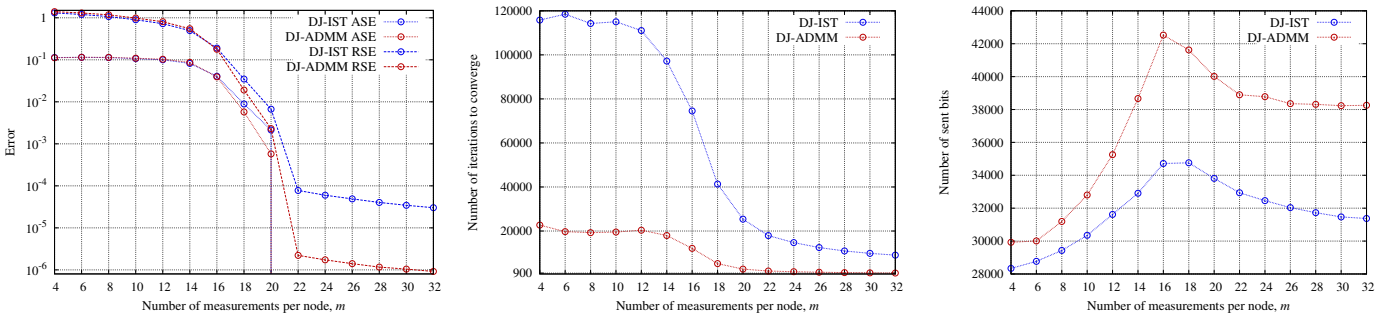


Figure 7: DJ-IST vs DJ-ADMM: ASE, RSE, number of iterations, and sent bits.

[13] J. Blanchard, M. Cermak, D. Hanle, and Y. Jing, "Greedy algorithms for joint sparse recovery," *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1694–1704, April 2014.

[14] Q. Ling and Z. Tian, "Decentralized support detection of multiple measurement vectors with joint sparsity," in *IEEE ICASSP*, 2011, pp. 2996–2999.

[15] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[16] D. Sundman, S. Chatterjee, and M. Skoglund, "Distributed greedy pursuit algorithms," *Signal Processing*, vol. 105, pp. 298–315, 2014.

[17] S. M. Fosson, J. Matamoros, C. Antón-Haro, and E. Magli, "Distributed support detection of jointly sparse signals," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6434–6438.

[18] T. Wimalajeewa and P. Varshney, "OMP based joint sparsity pattern recovery under communication constraints," *IEEE Trans. Signal Process.*, vol. 62, no. 19, pp. 5059–5072, Oct 2014.

[19] E. J. Candès, M. B. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journ. Fourier Anal. Appl.*, vol. 14, no. 5-6, pp. 877–905, 2008.

[20] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *IEEE J. Sel. Top. Signal Process.*, vol. 4, no. 2, pp. 317–329, April 2010.

[21] X. Chen and W. Zhou, "Convergence of the reweighted ℓ_1 minimization algorithm for ℓ_2-ℓ_p minimization," *Computational Optimization and Applications*, vol. 59, no. 1-2, pp. 47–61, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10589-013-9553-8

[22] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *J. Amer. Statist. Assoc.*, vol. 96, no. 456, pp. 1348 – 1360, 2001.

[23] P. Zhao and B. Yu, "On model selection consistency of Lasso," *J. Mach. Learn. Res.*, vol. 7, pp. 2541 – 2563, 2006.

[24] M. J. Wainwright, "Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (Lasso)," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2183–2202, 2009. [Online]. Available: http://dx.doi.org/10.1109/TIT.2009.2016018

[25] J. Fan and J. Lv, "Nonconcave penalized likelihood with np-dimensionality," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5467 – 5484, Aug 2011.

[26] J. Fan, L. Xue, and H. Zou, "Strong oracle optimality of folded concave penalized estimation," *Ann. Statist.*, vol. 42, no. 3, pp. 819 – 849, 06 2014. [Online]. Available: http://dx.doi.org/10.1214/13-AOS1198

[27] H. Zou and R. Li, "One-step sparse estimates in nonconcave penalized likelihood models," *Annals of Statistics*, vol. 36, no. 4, p. 1509, 2008.

[28] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *Ann. Statist.*, vol. 38, no. 2, pp. 894 – 942, 2010. [Online]. Available: http://www.jstor.org/stable/25662264

[29] C.-H. Zhang and T. Zhang, "A general theory of concave regularization for high-dimensional sparse estimation problems," *Statist. Sci.*, vol. 27, no. 4, pp. 576 – 593, 11 2012. [Online]. Available: http://dx.doi.org/10.1214/12-STS399

[30] G. Gasso, A. Rakotomamonjy, and S. Canu, "Recovering sparse signals with a certain family of nonconvex penalties and dc programming," *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4686 – 4698, 2009.

[31] M. Fazel, H. Hindi, and S. P. Boyd, "Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices," in *American Control Conference, 2003. Proceedings of the 2003*, vol. 3. IEEE, 2003, pp. 2156 – 2162.

[32] A. Antoniadis and J. Fan, "Regularization of wavelet approximations," *J. Amer. Statist. Assoc.*, vol. 96, no. 45, pp. 939 – 967, 2011.

[33] B. Aragam and Q. Zhou, "Concave penalized estimation of sparse gaussian bayesian networks," *J. Mach. Learn. Res.*, vol. in press, 2015.

[34] M. Fornasier, "Numerical methods for sparse recovery," in *Theoretical Foundations and Numerical Methods for Sparse Recovery*, M. Fornasier, Ed. Radon Series Comp. Appl. Math., de Gruyter, 2010, pp. 93–200.

[35] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *J. Fourier Anal. Appl.*, vol. 14, no. 5-6, pp. 629 – 654, 2008.

[36] S. Foucart, "Hard thresholding pursuit: An algorithm for compressive sensing," *SIAM J. Numer. Anal.*, vol. 49, no. 6, pp. 2543–2563, 2011.

[37] A. Kyrillidis and V. Cevher, "Recipes on hard thresholding methods," in *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2011, pp. 353–356.

[38] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1 – 122, 2010.

[39] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing," *SIAM J. Sci. Comp.*, vol. 33, no. 1, pp. 250–278, 2011.

[40] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Math. Program., Ser. A*, vol. 146, no. 1, pp. 459–494, 2013.

[41] A. M. Bagirov, L. Jin, N. Karmitsa, A. Al Nuaimat, and N. Sultanova, "Subgradient method for nonconvex nonsmooth optimization," *J. Optim. Theory Appl.*, vol. 157, no. 2, pp. 416–435, 2013.

[42] X. Chen, "Smoothing methods for nonsmooth, nonconvex minimization," *Math. Program. Ser. B*, vol. 134, no. 1, pp. 71–99, 2012.

[43] J. V. Burke, A. S. Lewis, and M. L. Overton, "A robust gradient sampling algorithm for nonsmooth, nonconvex optimization," *SIAM J. Optim.*, vol. 15, no. 3, pp. 751–779, 2005.

[44] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[45] R. J. Tibshirani, "The Lasso problem and uniqueness," *Electronic Journal of Statistics*, vol. 7, pp. 1456–1490, 2013.

[46] M. Hong and Z. Luo, "On the linear convergence of the alternating direction method of multipliers," *arXiv preprint arXiv:1208.3922*, 2012.

[47] J. Matamoros, S. M. Fosson, E. Magli, and C. Antón-Haro, "Distributed ADMM for in-network reconstruction of sparse signals with innovations," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 429 – 433.

[48] ——, "Distributed ADMM for in-network reconstruction of sparse signals with innovations," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 4, pp. 225 – 234, 2015.

[49] C. Ravazzi, S. M. Fosson, and E. Magli, "Distributed iterative thresholding for ℓ_0/ℓ_1-regularized linear inverse problems," *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 2081 – 2100, 2015.

[50] ——, "Energy-saving gossip algorithm for compressed sensing in multi-agent systems," in *Proc. of IEEE ICASSP*, 2014, pp. 5060 – 5064.

[51] ——, "Randomized algorithms for distributed nonlinear optimization under sparsity constraints," *IEEE Trans. Signal Process. (to appear)*, 2015.