A GREEDY PURSUIT ALGORITHM FOR DISTRIBUTED COMPRESSED SENSING

Dennis Sundman, Saikat Chatterjee, Mikael Skoglund

School of Electrical Engineering and ACCESS Linnaeus Centre
KTH - Royal Institute of Technology, SE-10044 Stockholm, Sweden
denniss@kth.se, sach@kth.se, skoglund@kth.se

ABSTRACT

We develop a greedy pursuit algorithm for solving the distributed compressed sensing problem in a connected network. This algorithm is based on subspace pursuit and uses the mixed support-set signal model. Through experimental evaluation, we show that the distributed algorithm performs significantly better than the stand-alone (disconnected) solution and close to a centralized (fully connected to a central point) solution.

Index Terms— Distributed compressed sensing, greedy pursuit.

1. INTRODUCTION

Compressed sensing (CS) [1, 2] refers to a class of under-sampling problems, where the measured data is inherently sparse. In the general CS problem, we collect few sample points and endeavor for reconstructing a larger sparse signal. CS has been shown potentially useful in a wide range of applications, for example brain-scanning [3], spectrum estimation [4] and sensor networks [5]. Currently, three main classes of CS reconstruction algorithms are in practice: convex relaxation, non-convex and greedy-pursuit.

Convex relaxation algorithms are theoretically elegant and they provide optimal performance, but at the expense of higher computational complexity. On the other hand, greedy-pursuit algorithms have shown to provide good performance at low complexity. Examples of such algorithms are subspace pursuit (SP) [6] and orthogonal matching pursuit (OMP) [7]. From a measurement vector, the main principle of greedy-pursuit algorithms is to estimate the underlying support-set of a sparse vector followed by evaluating the associated signal values. The support set is the set of indices corresponding to the non-zero elements of a sparse vector. To estimate the support set and the associated signal values, the greedy-pursuit algorithms use linear algebraic tools, for example the matched filter for detection and least squares for estimation.

In the literature, CS is presumably considered for a setup where all the measurements are acquired from a single sensor. While substantial amounts of work have been put for the single sensor CS problem, not much efforts have been put for a distributed CS reconstruction problem. A distributed CS setup uses multiple sensors to collect the measurements of sparse signals sharing some common information, but the sensors are not necessarily centrally connected to a fusion center. There are some results for the distributed CS problem based on convex relaxation algorithms [4, 8]. To the best of the authors knowledge, there exists no solution for efficiently solving the distributed CS problem based on greedy-pursuit algorithms.

This work was supported in part by the Swedish Research Council and by VINNOVA.

We endeavor for constructing a de-centralized, distributed CS algorithm based on greedy-pursuits. Using the signal model of [9] and the computationally efficient subspace pursuit (SP) algorithm, we develop a decentralized algorithm called distributed SP (DiSP). The signal model of [9] is called the mixed support-set model and can be used to describe joint sparsity information of several sparse signals under acquisition at different sensor nodes in a connected network. In a connected network, the DiSP algorithm first finds an estimate of the local support-set and exchange this local estimate to its neighbors over the network. Using the support-set estimates received from all the neighbors, the sensor node attempts to find a better estimate of the support-set. This iterative strategy of exchanging support-set estimates continues until convergence is achieved over the distributed network. By experimental evaluation, we show that the DiSP provides a substantial increase of reconstruction performance (for a moderately connected network), almost as good as a centralized (fully connected to a central point) solution.

Notations: Let a matrix be denoted as $\mathbf{A} \in \mathbb{R}^{M \times N}$ and a vector as $\mathbf{x} \in \mathbb{R}^N$. $\mathcal{T}$ is the support-set of $\mathbf{x}$, which is defined in the next section. $\mathbf{A}_{\mathcal{T}}$ is the sub-matrix consisting of the columns in $\mathbf{A}$ corresponding to the elements in a set $\mathcal{T}$. Similarly $\mathbf{x}_{\mathcal{T}}$ is a vector formed by the components of $\mathbf{x}$ that are indexed by $\mathcal{T}$. The pseudo inverse of $\mathbf{A}$ is denoted as $\mathbf{A}^\dagger$ and the matrix transpose as $\mathbf{A}^T$.

2. DISTRIBUTED COMPRESSED SENSING

Using a general multiple sensor node system setup [10], we first describe the distributed CS problem and then the mixed support-set signal model. The mixed support-set model was introduced in [9] where it is shown to be a generalization over previous signal models proposed in [10, 11, 12]. We also mention network topology and provide some algorithmic notations.

For the l'th sensor, we have the sparse signal $\mathbf{x}_l \in \mathbb{R}^N$ which is observed for the distributed CS problem as

$$\mathbf{y}_l = \mathbf{A}_l \mathbf{x}_l + \mathbf{w}_l, \qquad \forall l \in \{1, 2, ..., L\}, \tag{1}$$

where $\mathbf{y}_l \in \mathbb{R}^M$ is a measurement vector, $\mathbf{A}_l \in \mathbb{R}^{M \times N}$ a measurement matrix, $\mathbf{w}_l \in \mathbb{R}^M$ is the measurement error, $M < N$. $\mathbf{A}_l$ and $\mathbf{w}_l$ are independent across l. The signal vector $\mathbf{x}_l$ has K_l non-zero components with indices $\mathcal{T}_l = \{i : x_l(i) \neq 0\}$. $\mathcal{T}_l$ is referred to as the support-set of $\mathbf{x}_l$ with cardinality $|\mathcal{T}_l| = \|\mathbf{x}_l\|_0 = K_l$.

The distributed CS reconstruction problem endeavors for finding $\mathbf{x}_l$ for all l by exploiting some shared structure among the l sensors defined by the underlying signal model and by exchanging some information over the given network topology.

2.1. The Mixed Support-Set Model

Now, we describe the mixed support-set signal model with a shared structure where the signal vector $\mathbf{x}_l$ consists of two parts

$$\mathbf{x}_l = \mathbf{z}_l^{(c)} + \mathbf{z}_l^{(p)}, \quad \forall l \in \{1, 2, ..., L\}. \tag{2}$$

In (2) both $\mathbf{z}_l^{(c)}$ and $\mathbf{z}_l^{(p)}$ have independent non-zero components. There are $K_l^{(p)}$ non-zero values associated with $\mathbf{z}_l^{(p)}$. For simplicity we assume that the non-zero values are located uniformly at random over the support-set $\mathcal{T}_l^{(p)} \in \{1, 2, \ldots, N\}$, and $\forall l \in \{1, 2, \ldots, L\}$. For $\mathbf{z}_l^{(c)}$ there are similarly $K^{(c)}$ non-zero components with the constraint that the associated support-set $\mathcal{T}_l^{(c)}$ is shared, as $\mathcal{T}_l^{(c)} = \mathcal{T}^{(c)}, \forall l \in \{1, 2, \ldots, L\}$. The elements of $\mathcal{T}^{(c)}$ are the same (common) to all signals, but unknown to the re-constructor[1]. For the l'th node, this gives a support-set $\mathcal{T}_l$ for each $\mathbf{x}_l$ as

$$\mathcal{T}_l = \mathcal{T}^{(c)} \cup \mathcal{T}_l^{(p)}, \quad \forall l \in \{1, 2, ..., L\}. \tag{3}$$

We define $K_{l,\max} = |\mathcal{T}^{(c)}| + |\mathcal{T}_l^{(p)}| = K^{(c)} + K_l^{(p)}$. Note that the support-sets can intersect, so $K_{l,\max} \geq K_l$.

2.2. Network Topology

The DiSP algorithm presented here can work in any network topology. Having that said, we can expect that the algorithm will perform better (faster convergence, better result, etc.) if the network is well connected. The effects of the network topology on the performance of DiSP is not considered in this paper. Here, we focus on the development of the DiSP algorithm for any given network.

2.3. Algorithmic Notation

For clarity in the algorithmic notation, we define three algorithm functions as follows

$$\texttt{resid}(\mathbf{y}, \mathbf{B}) \triangleq \mathbf{y} - \mathbf{B}\mathbf{B}^{\dagger}\mathbf{y}, \text{ for some matrix } \mathbf{B}, \tag{4}$$

$$\texttt{max\_indices}(\mathbf{x}, k) \triangleq \{\text{the set of indices corresponding to the } k \text{ largest amplitude components of } \mathbf{x}\}, \tag{5}$$

and

$$\texttt{add}_1(\mathbf{s}, \mathcal{T}) \triangleq \{\forall j \in \mathcal{T}, \textit{ perform } s_j = s_j + 1\}, \tag{6}$$

where $\mathbf{s} = [s_1 \ s_2 \ \ldots s_N]$ and $s_j \geq 0$.

We refer to the node executing the algorithm as the present or current node. The current node can be identified by sub-index l^*. We also define $\mathcal{L}_{l^*}$ as the set of neighboring nodes connected to the current node, including itself.

3. DISTRIBUTED SUBSPACE PURSUIT

To implement the distributed subspace pursuit (DiSP) algorithm, we first have to modify the standard SP algorithm presented in Algorithm 2. Using the modified SP by calling SP(.), the DiSP algorithm is carried out at each node, shown in Algorithm 1. The DiSP performs two main tasks: the communication task and the calculation task. For the communication task, some underlying protocol needs to be present that takes care of the transmissions in the network. For the calculation task, the modified SP and the common support-set estimation are the main parts carried out by the algorithm. We briefly mention the complexity of DiSP in subsection 3.2.

Algorithm 1 : Distributed SP (DiSP)
Executed in the l^*-th node, where $\mathcal{L}_{l^*}$ is the set of neighboring (Note that $l^* \in \mathcal{L}_{l^*}$)

Input: $\mathbf{A}_{l^*}$, $\mathbf{y}_{l^*}$, $K_{l^*}^{(p)}$, $K^{(c)}$
Initialization:
1: $K_{l^*,\max} = K_{l^*}^{(c)} + K^{(p)}$
2: $(\hat{\mathcal{T}}_{l^*}, \hat{\mathbf{x}}_{l^*}, n_{l^*}) \leftarrow \text{SP}(\mathbf{A}_{l^*}, \mathbf{y}_{l^*}, \emptyset, K_{l^*,\max})$
3: $n_{l^*}^{\text{old}} \leftarrow n_{l^*}$
4: $\hat{\mathcal{T}}_l \leftarrow \emptyset, \ \forall\, l \in \mathcal{L}_{l^*} \setminus l^*$ (i.e. except l^*)
Iteration:
1: **repeat**
2: **if** $n_{l^*} > n_{l^*}^{\text{old}}$ **then**
3: $(\hat{\mathcal{T}}_{l^*}, \hat{\mathbf{x}}_{l^*}, n_{l^*}) \leftarrow (\hat{\mathcal{T}}_{l^*}^{\text{old}}, \hat{\mathbf{x}}_{l^*}^{\text{old}}, n_{l^*}^{\text{old}})$
4: **end if**
5: $(\hat{\mathcal{T}}_{l^*}^{\text{old}}, \hat{\mathbf{x}}_{l^*}^{\text{old}}, \eta_{l^*}^{\text{old}}) \leftarrow (\hat{\mathcal{T}}_{l^*}, \hat{\mathbf{x}}_{l^*}, \eta_{l^*})$
6: $\hat{\mathcal{T}}_l^{\text{old}} \leftarrow \hat{\mathcal{T}}_l, \ \forall\, l \in \mathcal{L}_{l^*} \setminus l^*$
7: { **Communication:** Send $\hat{\mathcal{T}}_{l^*}$ to all nodes $l \in \mathcal{L}_{l^*}$ }
8: { **Communication:** Receive $\hat{\mathcal{T}}_l$ from all nodes $l \in \mathcal{L}_{l^*}$ }
9: $\mathbf{s}_{l^*} \leftarrow \mathbf{0}_{N\times 1}$
10: **for** each $l \in \mathcal{L}_{l^*}$ **do**
11: $\mathbf{s}_{l^*} \leftarrow \texttt{add}_1(\mathbf{s}_{l^*}, \hat{\mathcal{T}}_l)$
12: **end for**
13: $\hat{\mathcal{T}}_{l^*}^{(c)} \leftarrow \texttt{max\_indices}(\mathbf{s}_{l^*}, K^{(c)})$
14: $(\hat{\mathcal{T}}_{l^*}, \hat{\mathbf{x}}_{l^*}, n_{l^*}) \leftarrow \text{SP}(\mathbf{A}_{l^*}, \mathbf{y}_{l^*}, \hat{\mathcal{T}}_{l^*}^{(c)}, K_{l^*,\max})$
15: **until** $(n_{l^*} \geq n_{l^*}^{\text{old}})$ **and** $(\hat{\mathcal{T}}_l = \hat{\mathcal{T}}_l^{\text{old}} \ \forall\, l \in \mathcal{L}_{l^*})$
Output: $\hat{\mathbf{x}}_{l^*}$, $\hat{\mathcal{T}}_{l^*}$

Input to the algorithm is the present node's sensing matrix $\mathbf{A}_{l^*}$, the size of the private and common support set $K_{l^*}^{(p)}$ and $K^{(c)}$, and the present node's measurement vector $\mathbf{y}_{l^*}$. For the initialization phase of the algorithm, before any communication has taken place, a standard SP is run to achieve a first estimate of the current node's support-set and the signal. We also store the residual norm as n, to use as performance measure and set some initial parameter values for the algorithm to start properly.

For the iteration phase, there are four main functionalities being processed. **(i)** Step 2 to 4 makes sure that the result do not deviate away from the best solution, which could happen if the estimated support set in step 14 is bad. **(ii)** In step 7 to 8 we have the communication phase, where the support-set data are exchanged among the nodes. Assuming the measurement-matrices $\mathbf{A}_l$ are known, the only exchanged information is the current estimates of the support-sets $\hat{\mathcal{T}}_l$. **(iii)** In step 9 to 13, an estimate of the common support-set is achieved. We mention that if the current node has no neighbors, $\hat{\mathcal{T}}_{l^*}^{(c)}$ will be chosen as the first $K^{(c)}$ components of $\hat{\mathcal{T}}_{l^*}$. **(iv)** In step 15, the convergence criterion is chosen. We propose to stop when the residual norm is no longer decreasing and no new data is coming in. The first, residual norm, criterion comes naturally from the SP algorithm, but as long as new support-set data is coming in, we may improve the result in a later iteration. Thus, the second criterion is added to make sure the algorithm does not stop until the node experience a stable situation, with no additional support-set information received.

[1] For easy practical implementation, we assume that the elements are uniformly distributed over $\mathcal{T}^{(c)}$ and $\mathcal{T}_l^{(p)}$.

3.1. Modified Subspace Pursuit

As mentioned in the description of DiSP, we here describe the modified SP in Algorithm 2. The initialization phase has, compared to the standard SP, been modified so that it can use an initial support-set. The modified SP reduces to the standard SP as defined by Dai and Milenkovic [6] when $\mathcal{T}_{\text{ini}} = \emptyset$. The sub-index l which describes what node a particular variable belongs to is not present in this algorithm because the modified SP has no knowledge about any networks. All variables belong to the current node.

Algorithm 2 : modified SP

Input: $\mathbf{A}$, $\mathbf{y}$, $\mathcal{T}_{\text{ini}}$, $K_{\max}$

Initialization:

1: $\mathcal{T}' \leftarrow \texttt{max\_indices}\left(\mathbf{A}^T\mathbf{y}, K_{\max}\right) \cup \mathcal{T}_{\text{ini}}$
2: $\hat{\mathbf{x}}$ such that $\hat{\mathbf{x}}_{\mathcal{T}'} = \mathbf{A}_{\mathcal{T}'}^{\dagger}\mathbf{y}$ and $\hat{\mathbf{x}}_{\bar{\mathcal{T}}'} = \mathbf{0}$
3: $\mathcal{T}_0 \leftarrow \texttt{max\_indices}(\hat{\mathbf{x}}, K_{\max})$
4: $\mathbf{r}_0 \leftarrow \texttt{resid}(\mathbf{y}, \mathbf{A}_{\mathcal{T}_0})$
5: $k \leftarrow 0$

Iteration:

1: **repeat**
2: $k \leftarrow k + 1$
3: $\mathcal{T}' \leftarrow \mathcal{T}_{k-1} \cup \texttt{max\_indices}\left(\mathbf{A}^T\mathbf{r}_{k-1}, K_{\max}\right)$
4: $\hat{\mathbf{x}}$ such that $\hat{\mathbf{x}}_{\mathcal{T}'} = \mathbf{A}_{\mathcal{T}'}^{\dagger}\mathbf{y}$ and $\hat{\mathbf{x}}_{\bar{\mathcal{T}}'} = \mathbf{0}$
5: $\mathcal{T}_k \leftarrow \texttt{max\_indices}(\hat{\mathbf{x}}, K_{\max})$
6: $\mathbf{r}_k \leftarrow \texttt{resid}(\mathbf{y}, \mathbf{A}_{\mathcal{T}_k})$
7: **until** $(\|\mathbf{r}_k\|_2 \geq \|\mathbf{r}_{k-1}\|_2)$
8: $k \leftarrow k - 1$ ('Previous iteration count')

Output:

1: $\hat{\mathcal{T}} \leftarrow \mathcal{T}_k$
2: $\hat{\mathbf{x}}$ such that $\hat{\mathbf{x}}_{\mathcal{T}_k} = \mathbf{A}_{\mathcal{T}_k}^{\dagger}\mathbf{y}$ and $\hat{\mathbf{x}}_{\bar{\mathcal{T}}_k} = \mathbf{0}$
3: $n \leftarrow \|\mathbf{r}_k\|_2$

At k'th iteration stage, the modified SP algorithm forms the matched filter $\mathbf{A}^T\mathbf{r}_{k-1}$, identifies the indices corresponding to the $K_{\max}$ largest amplitudes followed by joining with the old support-set (step 3). This support-set $\mathcal{T}'$ is likely to be bigger than $K_{\max}$. The algorithm solves a least squares problem with the selected indices of $\mathcal{T}'$ and identifies the new indices corresponding to the $K_{\max}$ largest amplitudes (step 4 and 5) followed by finding the residual (step 6). This process is repeated until the residual norm is non-decreasing. In addition with the sparse signal estimate $\hat{\mathbf{x}}$, we also output the estimated support-set $\hat{\mathcal{T}}$ and the final residual norm.

3.2. Complexity of DiSP

The modified SP requires one matched filter and two orthogonal projections in each iteration. Therefore its complexity is approximately $\mathcal{O}(KMN)$ [13]. For the DiSP, the modified SP is called once for every iteration which continues to iterate until convergence is achieved. We will refer to this convergence by the parameter γ which depends on the network topology, noise level, total number of nodes and size of the common support-set. Thus the convergence for DiSP is approximately $\mathcal{O}(\gamma KMN)$. For the system setup in the simulation results of section 4, the number of iterations turned out to be less than ten ($\gamma < 10$).

4. SIMULATIONS AND RESULTS

In the simulations we are interested in how close DiSP comes to the centralized solution, referred to as joint SP (JSP) developed in [9], and how the DiSP compares to the standard SP (with no network).

We report the results for clean and noisy measurement cases. In the noisy case we have chosen signal-to-measurement-noise-ratio (SMNR) 20 dB, (i.e., $10\log_{10}\frac{\mathbb{E}\{\|\mathbf{x}\|_2^2\}}{\mathbb{E}\{\|\mathbf{w}\|_2^2\}} = 20$). Note that we drop the subscript l because we are averaging over all nodes l. To compare the algorithms, the performance measure chosen is the signal-to-reconstruction-error-ratio (SRER) which is defined as

$$\text{SRER} = 10\log_{10}\frac{\mathbb{E}\left\{\|\mathbf{x}\|_2^2\right\}}{\mathbb{E}\left\{\|\mathbf{x}-\hat{\mathbf{x}}\|_2^2\right\}}. \tag{7}$$

Next we describe the simulation setup. We emulate a connected network with a binary connection matrix $\mathbf{C}$, where a "1" in c_{ij} corresponds to a connection from node i to node j. In any CS setup, all sparse signals are expected to be exactly reconstructed if the number of measurements are more than a certain threshold value. The computational complexity to test this uniform reconstruction ability is exponentially high. Instead, we can rely on empirical testing, where SRER is computed for random measurement matrix ensemble. We define the fraction of measurements as

$$\alpha = \frac{M}{N}. \tag{8}$$

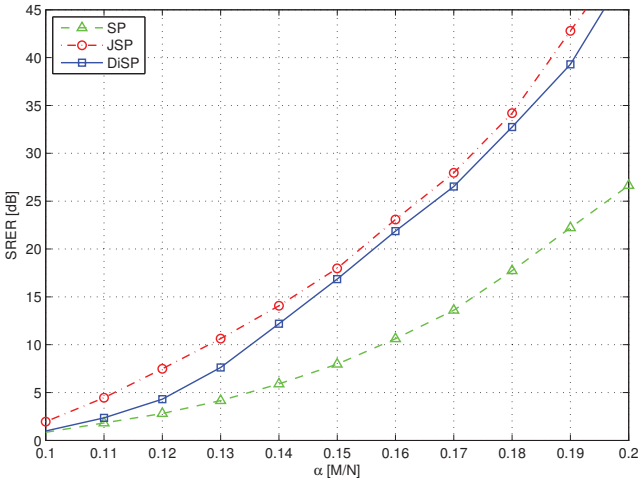
Using α, the steps of testing strategy is as follows:

1. Given the signal parameter N and connection matrix $\mathbf{C}$, choose an α (such that M is an integer).
2. Randomly generate:
   - A set of $M \times N$ sensing matrices $\{\mathbf{A}_l\}_{l=1}^L$ where the components are drawn from an i.i.d. Gaussian source (i.e. $a_{m,n} \sim \mathcal{N}\left(0, \frac{1}{M}\right)$) and scale the columns of $\mathbf{A}_l$ to unit-norm.
   - Support-sets $\mathcal{T}^{(c)}$ and $\{\mathcal{T}_l^{(p)}\}_{l=1}^L$ of cardinality $K^{(c)}$ and $\{K_l^{(p)}\}_{l=1}^L$, respectively. The support-sets are uniformly chosen from $\{1, 2, ..., N\}$.
   - A set of signal vectors $\{\mathbf{x}_l\}_{l=1}^L$ following (2), where $\{\mathbf{z}_l^{(c)}\}_{l=1}^L$ and $\{\mathbf{z}_l^{(p)}\}_{l=1}^L$ corresponding to the non-zero components (support-sets determined in step 2). The non-zero components in the vectors are chosen i.i.d from a Gaussian source.
3. Compute the measurements $\mathbf{y}_l = \mathbf{A}_l\mathbf{x}_l + \mathbf{w}_l, \forall l \in \{1, 2, ..., L\}$. Here $\mathbf{w}_l \sim \mathcal{N}(\mathbf{0}, \sigma_l^2\mathbf{I}_M)$.
4. Apply the CS algorithms on the data $\{\mathbf{y}_l\}_{l=1}^L$. The connection matrix $\mathbf{C}$ is used to determine how to distribute the data in the network.
5. We let the distributed CS algorithm run until no node in the network improve any more.

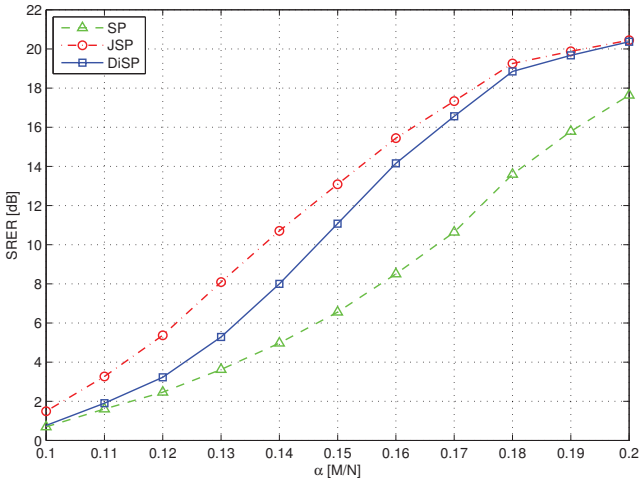
In the simulation procedure above, for each node $l \in \{1, 2, \ldots, L\}$, Q sets of sensing and connection matrices are created. For each node and each sensing matrix, P sets of data vectors are created. In total, we will average over $L \cdot Q \cdot P$ data to evaluate the performance.

4.1. Simulation Results

For the plots presented in this paper, we have chosen: one fixed connection-matrix $\mathbf{C}$ such that each node is connected to exactly two other nodes (i.e., $c_{i,i} = c_{i,i+1} = 1$ and $c_{L,1} = 1$); $N = 500$; $K^{(c)} = 10$; and $\forall l, K_l^{(p)} = K^{(p)} = 10$. We have chosen nodes



(a) Clean measurements



(b) Noisy measurements, where SMNR = 20 dB

Fig. 1: SRER versus fraction of measurements for **clean** and **noisy** (SMNR $= 20$ dB) measurements.

$L = 10$ for which we have chosen number of $\mathbf{A}_l$'s to 100 (i.e. $Q = 100$) and the number of data-sets $\mathbf{x}$ to 100 (i.e. $P = 100$), giving a total number of $L \cdot Q \cdot P = 100000$ data for statistics.

In the figures there are three algorithms compared, DiSP, JSP and SP. SP is the standard subspace pursuit and JSP is a centralized version of the decentralized DiSP algorithm. For the SRER of a clean signal in Fig. 1a, we notice that DiSP performs significantly better than SP. At $\alpha = 0.16$ the gain is about 12 dB. For the SRER of noisy measurements (20 dB noise) in Fig. 1b, the improvement of performance is almost 6 dB at $\alpha = 0.16$. We also notice that DiSP provides close performance to the JSP.

In the simulation results shown here, the experiments were conducted using one distributed network topology $\mathbf{C}$ of limited connectivity as mentioned in the testing strategy. We have also results for various other network topologies and have achieved encouraging results, but do not show those results here because of space limitations.

5. CONCLUSION

In this paper, a distributed greedy pursuit algorithm called DiSP (distributed subspace pursuit) is developed. Using the mixed support-set model, it is shown how this algorithm can be developed based on the underlying standard subspace pursuit algorithm. By experimental evaluation in a well connected network, we conclude that the DiSP significantly improves the performance compared to the standard SP and comes close to the performance of a centralized (fully connected) solution.

6. REFERENCES

[1] D.L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289 –1306, April 2006.

[2] E.J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489 – 509, Feb. 2006.

[3] J.W. Phillips, R.M. Leahy, and J.C. Mosher, "Meg-based imaging of focal neuronal current sources," *IEEE Trans. Med. Imaging*, vol. 16, no. 3, pp. 338 –348, June 1997.

[4] F. Zeng, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multihop cognitive networks," *IEEE Journal Selected Topics in Signal Processing*, vol. PP, no. 99, pp. 1 –1, 2010.

[5] A.Y. Yang, M. Gastpar, R. Bajcsy, and S.S. Sastry, "Distributed sensor perception via sparse representation," *Proc. of the IEEE*, vol. 98, no. 6, pp. 1077 –1088, June 2010.

[6] Wei Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2230 –2249, May 2009.

[7] J.A. Tropp and A.C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655 –4666, Dec. 2007.

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

[9] D. Sundman, S. Chatterjee, and M. Skoglund, "Greedy pursuits for compressed sensing of jointly sparse signals," in *Proc. Eur. Sig. Proc. Conf.*, Aug 2011.

[10] M.F. Duarte, S. Sarvotham, D. Baron, M.B. Wakin, and R.G. Baraniuk, "Distributed compressed sensing of jointly sparse signals," *Proc. Asilomar Conf. Signals, Sys., and Comp.*, pp. 1537 – 1541, Oct. 2005.

[11] S.F. Cotter, B.D. Rao, Kjersti Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Trans. Signal Processing*, vol. 53, no. 7, pp. 2477 – 2488, Jul. 2005.

[12] R. Gribonval, H Rauhut, K. Schnass, and P Vandergheynst, "Atoms of all channels, unite! Average case analysis of multi-channel sparse recovery using greedy algorithms," *J. Fourier Anal. and Appl.*, vol. 14, no. 5, pp. 655–687, 2008.

[13] S. Chatterjee, D. Sundman, M. Vehkaperä, and M. Skoglund, "Projection-based and look ahead strategies for atom selection," *IEEE Trans. Signal Processing*, vol. PP, no. 99, pp. 1, 2011.