DISTRIBUTED LINEAR BLIND SOURCE SEPARATION OVER WIRELESS SENSOR NETWORKS WITH ARBITRARY CONNECTIVITY PATTERNS

S.R Mir Alavi and W. Bastiaan Kleijn

School of Engineering and Computer Science, Victoria University of Wellington, New Zealand

ABSTRACT

Broad areal coverage and low cost make wireless sensor networks natural platforms for blind source separation (BSS). In this context, distributed processing is attractive because of low power requirements and scalability. However, existing distributed BSS algorithms either require a fully connected pattern of connectivity or require a high computational load at each sensor node. We introduce a distributed robust BSS algorithm that uses a fully shared computation and can be applied over any connected graph. This enables us to facilitate a low computational load at each node as well as low data transmission rates. Comparative experimental results confirm the effectiveness of the new method.

***Index Terms*—** Distributed Processing, Blind Source Separation, Sparse, Network, Bi-ADMM

1. INTRODUCTION

Communication is performed by transmitting signals, such as acoustic signals and radio signals, through a medium. It is common that signals originating from different sources are mixed in the transport medium before they are received. Hence, there is a need to extract the original source signals from a mixture of such source signals. The operation of separating source signals without prior information about the sources is referred to as blind source separation (BSS), e.g., [1].

Wireless sensor networks (WSN) are a rapidly developing technology, e.g., [2], and they form a natural platform for effective, low-cost BSS. In contrast to wired networks of sensors, WSNs facilitate a good coverage of an area where signals of interest are present at a low deployment cost.

In the context of sensor networks, research on blind source separation can be divided into centralized and de-centralized approaches. In centralized approaches, e.g., [3, 4], the recordings of all sensors (microphones) are transmitted to a centralized processor, the fusion center, to perform the separation procedure. Drawbacks of the centralized approach include the lack of scalability, high power consumption of the transmission and the need for hardware that can transmit over long distances [5].

De-centralized, in-network, processing approaches have the potential to be scalable and have a low power consumption [6, 7]. A simple form of decentralized BSS was discussed in [8, 9], where the separation procedure is pararallelized over a number of processors. However, the methods in [8, 9] retain the need to aggregate the observations of all sensors in each individual node. This imposes the need for a full connectivity pattern over the graph of the network and, hence, restricts the scalability of the BSS algorithm. A more scalable distributed algorithm is introduced in [10], but at the cost of a significantly higher computational load at each node.

In this paper, we implement a distributed form of a suitable BSS algorithm. The BSS algorithm is based on [11] with an extension based on [12]. A motivation for using an algorithm based on [11] is that it can handle both over-determined and under-determined cases (for the under-determined case regularization must be added). The extension based on [12] enables the BSS algorithm to be stable in situations where the number of sources is over-estimated.

Our distributed BSS algorithm consists of two steps for each time sample. In the first step it solves a constrained convex optimization problem to estimate the sources given the current observations and the current estimate of the mixing matrix. The objective function of the optimization problem forms a sum of N convex functions, each associated with an individual node of the network. In the second step, the mixing matrix is updated given the observation and the newly estimated sources. The second step is local and requires no communication across the nodes. The proposed distributed processing strategy enables the algorithm to be implemented on connected graphs of any connectivity pattern.

When our distributed BSS algorithm is applied for a well designed sparse network, its energy requirement for data transmission is low. An appropriate selection of the connections means that no transmission over long distances is required.

We consider the scenario in which each sensor node computes estimates of all source signals. Hence any node can send a source signal to its user. The new algorithm is particularly efficient if there are a low number of sources while there are higher number of sensor nodes.

We confirm the strong performance of the new distributed BSS with experiments for the case of delay-restricted (non-echoic) environments where the observations are obtained from the linear instantaneous mixture of the sources. The method is easily extended to apply to convolutional scenarios using conventional frequency domain approaches.

The remainder of this paper is organized as follows. In section 2 we formulate the linear BSS problem and introduce a suitable online learning algorithm to solve this problem. In section 3 we explain the proposed distributed solution. The evaluation of the proposed distributed-processing BSS algorithm is presented in section 4. It is followed by a conclusion in section 5.

2. LINEAR BLIND SOURCE SEPARATION

In this section, we describe the centralized algorithm that forms a suitable basis for a distributed BSS. The algorithm is aimed at low noise conditions and optimizes the mixing matrix to maximize the likelihood of the observed data. It is based on a linear generative model:

$$b_{N\times 1}(t) = A_{N\times M} s_{M\times 1}(t) + n_{N\times 1}(t), \qquad t = 1, ..., T, \quad (1)$$

where N is the number of microphones, M is the number of sources, $s(t)$ and $b(t)$ contain the sources and observed data at time index t respectively, $n(t)$ is a zero mean Gaussian noise and A is the mixing matrix.

A likelihood function for the mixing matrix can be written as [11]:

$$\mathrm{p}(b|A) = \int \mathrm{p}(b|s,A)\,\mathrm{p}(s)\mathrm{d}s. \tag{2}$$

In [11] a Gaussian $\mathrm{p}(b|s,A)$ and a fully factorizable prior Laplacian model $\mathrm{p}(s)$ are used. The Laplacianity of the prior makes the integral in (2) intractable. Hence an approximation of the logarithm of (2) is used as the objective function. The gradient ascent technique can then be used for the optimization. In [11] the natural gradient [13] ascent is used:

$$\begin{aligned} \triangle A &\propto AA^T \frac{\partial}{\partial A}\log \mathrm{p}(b|A) \\ \text{or} \quad A(t+1) &= A(t) - \mu A(t) F[y(t)], \end{aligned} \tag{3}$$

where $F[y(t)] = I - \psi[y(t)]y(t)^T$, $y(t) = A^{-1}(t)b(t)$ describes the estimated source signals at time index t, and ψ is the Marginal Score Function (MSF) defined as:

$$\begin{aligned} \psi[y] &= [\psi_1[y_1], ..., \psi_M[y_M]]^T \\ \psi_i[y_i] &= -\frac{d}{dy_i}\ln \mathrm{p}(y_i). \end{aligned} \tag{4}$$

For prior Laplacian models, $\psi_i[y_i]$ reduces to $\mathrm{sign}[y_i]$.

In the case of zero noise and $N = M$, (3) has been shown to have the same performance as the standard independent-component analysis (ICA) algorithm [14].

As there is no exact information about the variance of the source signals, the identity matrix is used as a prior assumption over the covariance matrix of the sources. This leads to instability if fewer sources than allowed for are present. To eliminate this problem the identity matrix is substituted with a dynamic matrix Λ in [12]:

$$F[y(t)] = \Lambda(t) - \psi[y(t)]y(t)^T, \tag{5}$$

where $\Lambda(t) = \mathrm{diag}[\mathrm{diag}[\psi[y(t)]y(t)^T]]$. This means that $F[y(t)]$ has a zero diagonal. In practice, (5) preserves the stability of the algorithm (3) when the number of sources is set too high. In the common situation where the number of sources is unknown a-priori, we can now allocate a source vector of relatively high dimension M. The algorithm will then extract up to M source signals if there are at least M sensor nodes in the network.

Algorithm 1 outlines the centralized processing version of our desired on-line linear BSS algorithm method for the case that $N \geq M$. A^+ denotes the pseudo-inverse of A.

Algorithm 1 An on-line centralized linear BSS Algorithm for a time sequence of T time samples. The algorithm updates the matrix $A(t)$ at each time sample t.

Initialize A
for $t = 1, ..., T$ **do**
  $y(t) = A^+(t)b(t)$
  $A(t+1) = A(t) - \mu A(t)(\Lambda(t) - \psi[y(t)]y(t)^T)$
end for

3. DISTRIBUTED PROCESSING APPROACH

In this section we discuss the distributed implementation of the algorithm outlined in section 2. First we introduce two different distributed algorithms that can be used for the first step of each iteration in section 3.1. This is followed by a description of the distributed implementation of the second step of the iteration in section 3.2.

3.1. Distributed processing of source estimation

The first iteration step of the Algorithm 1 estimates the source signals given the model parameters. That is, we have a model $A(t)$ and the observations $b(t)$ at time index t and want to find $y(t)$. We consider two different distributed processing solutions for this step: one of these benefits from a fusion center and the other does not.

3.1.1. Fusion-center (ADMM) approach

If $A(t)$ and $b(t)$ in (1) are known, then finding the estimate $y(t)$ of $s(t)$ can be formulated as the problem of finding the $y(t)$ that minimizes $\frac{1}{2}\|A(t)y(t) - b(t)\|_2^2$. We can split this objective function on a row-by-row basis by writing $f_i(y^{\{i\}}) = \frac{1}{2}\|A_i(t)y^{\{i\}} - b_i(t)\|_2^2$, where A_i and b_i are the i^{th} row of A and b respectively. The optimization problem can now be written as:

$$\min_y \sum_{i=1}^{N} f_i(y^{\{i\}}) \;\text{ s.t. }\; y^{\{i\}} - z = 0 \;\text{ for }\; i = 1, ..., N, \tag{6}$$

where $z \in \{(y^{\{1\}}, ..., y^{\{N\}}) | y^{\{1\}} = y^{\{2\}} = ... = y^{\{N\}}\}$.

The augmented Lagrangian function of the problem above can be written as:

$$L(y, z, u) = \sum_{i=1}^{N} \{f_i(y^{\{i\}}) + \frac{\rho}{2}||y^{\{i\}} - z + u_i||_2^2\} \tag{7}$$

Making use of the augmented Lagrangian in (7), the problem (6) can be solved by the well-known alternating direction method of multipliers (ADMM) [15, 16, 17] as follows:

$$\begin{aligned} y^{\{i,k+1\}} &= \arg\min_{y^{\{i\}}}(f_i(y^{\{i\}}) + \frac{\rho}{2}||y^{\{i\}} - z^k + u_i^k||_2^2) \\ z^{k+1} &= \frac{1}{N}\sum_{i=1}^{N}(y^{\{i,k+1\}} + u_i^k) \\ u_i^{k+1} &= u_i^k + y^{\{i,k+1\}} - z^{k+1}. \end{aligned} \tag{8}$$

The standard ADMM algorithm gathers the u_i and $y^{\{i\}}$ in the fusion node, calculates the average, and then broadcasts the results to each node of the network, all at each iteration of the ADMM algorithm. The left side of Fig. 1 shows an example graph of this fusion-center based approach.

3.1.2. De-centralized approach

To avoid the need for a fusion center, we can also use fully distributed optimization algorithms such as the distributed ADMM algorithm [18, 19, 20], or the Bi-ADMM algorithm [21], to find the $y(t)$ that minimizes $\frac{1}{2}\|A(t)y(t) - b(t)\|_2^2$. We use the Bi-ADMM algorithm as example because of its superior convergence properties.

The Bi-ADMM algorithm for a graphical model G=(ν, ε) with $|\nu| = N$ solves problems of the form:

$$\min_y \sum_{i\in\nu} f_i(y^{\{i\}}) \text{ s.t. } B_{i\to j}y^{\{i\}} + B_{j\to i}y^{\{j\}} = c_{ij}\,\forall(i,j) \in \varepsilon \tag{9}$$

where $B_{i\to j}$ and $B_{j\to i}$ specify linear constraints.

According to [21], the augmented primal-dual Lagrangian function of the problem (9) can be written as:

$$\begin{aligned} L_P(y, \lambda) = &\sum_{i\in\nu}[f_i(y^{\{i\}}) - \sum_{j\in N(i)} \lambda_{j|i}^T(B_{i\to j}y^{\{i\}}) - f_i^*(B_i^T\lambda_i)] \\ &+ \sum_{(i,j)\in\varepsilon}(\frac{1}{2}||y^{\{i\}} - y^{\{j\}}||_2^2 - \frac{1}{2}||\lambda_{i|j} - \lambda_{j|i}||_2^2) \end{aligned} \tag{10}$$

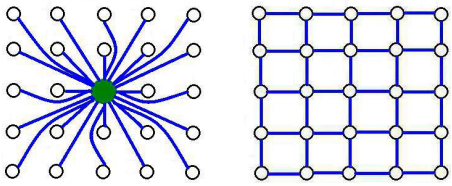

Fig. 1. Illustration of the connectivity patterns. Left: Fusion-center approach. Right: De-centralized approach.

where $f^*(.)$ is the conjugate function of $f(.)$, $\lambda_{i|j}$ is the variable which is held in node i and is related to node j and $B_i^T\lambda_i = \sum_{j\in N(i)} B_{i\to j}^T \lambda_{i|j}$.

If there is a solution y^* for the problem (9), there will be a λ^* such that (y^*, λ^*) is the saddle point of the augmented primal-dual Lagrangian function (10). Consequently the solution of the problem (9) can be obtained through the following optimization problem:

$$(y^*, \lambda^*) = \arg\min_{y}\max_{\lambda} L_P(y, \lambda) \tag{11}$$

For our BSS problem we define the $f_i(y^{\{i\}})$ as in section 3.1.1 and seek a consensus (equality) for $y^{\{i\}}, i = 1, ..., N$, which is enacted by setting $B_{i\to j} = -B_{j\to i} = I_M$, where I_M is the identity matrix of size M, and $c_{ij} = 0$.

For our case the Bi-ADMM algorithm leads to the following updates:

$$\begin{aligned} y^{\{i,k+1\}} =& \arg\min_{y^{\{i\}}}[\sum_{j\in N(i)} \frac{1}{2}||y^{\{i\}} - y^{\{j,k\}}||_2^2 \\ & - {y^{\{i\}}}^T(\sum_{j\in N(i)} \text{sign}(i-j)\lambda_{j|i}^k) + f_i(y^{\{i\}})] \\ \lambda_{i|j}^{k+1} =& \lambda_{j|i}^k + \text{sign}(i-j)(y^{\{j,k\}} - y^{\{i,k+1\}}) \end{aligned} \tag{12}$$

where the dual variable $\lambda_{i|j}$ is held in node i and is related to node j and $N(i)$ indicates the neighbours of node i.

Both synchronous and asynchronous updates schemes can be used to implement (12). BiADMM converges at a rate of $O(1/k)$ regardless of the graph topology. The right side of Fig. 1 shows an example graph for the de-centralized approach.

3.2. Distributed processing of the parameter update

The second step performed at each time sample by our distributed BSS algorithm updates the mixing matrix A. Using again the separated structure introduced in section 3.1, we treat $y^{\{i\}}(t)$, $A_i(t)$ and $b_i(t)$ as the data that are owned by the node i. We then use the mixing matrix update scheme of section 2, which can be carried in parallel on all nodes without the necessity for communications between the nodes; all required data are available at each individual node. Each node performs the following operation:

$$A_i(t+1) = A_i(t) - \mu A_i(t) F[y^{\{i,k\}}(t)], \tag{13}$$

where $F[y^{\{i,k\}}(t)] = \Lambda(t) - \psi[y^{\{i,k\}}(t)]y^{\{i,k\}}(t)^T$ and $\Lambda(t) = \text{diagdiag}[[\psi[y^{\{i,k\}}(t)]y^{\{i,k\}}(t)^T]]$.

This completes the algorithm. The entire algorithm is outlined in Algorithm 2 for the Bi-ADMM case. For the ADMM case the inner loop is replaced by the ADMM algorithm.

Algorithm 2 On-line distributed linear BSS Algorithm for a time sequence of T time samples. At each time t, the algorithm estimates the source samples as the consensus of $y^{\{i\}}$ given $A(t)$ and observations $b(t)$, and then updates the demixing matrix $A(t)$.

Initialize $A_i, i \in \nu$
for $t = 1, ..., T$ **do**
  Initialize $y^{\{i,1\}}(t) = 0, \lambda_{i|j}^1 = 0 \ \forall i, j \in \nu, \ k \leftarrow 1$
  while the stopping criteria is not met [in parallel] **do**
    $y^{\{i,k+1\}}(t) = \arg\min_{y^{\{i\}}} [\sum_{j\in N(i)} \frac{1}{2}||y^{\{i\}} - y^{\{j,k\}}(t)||^2$
    $-{y^{\{i\}}}^T(\sum_{j\in N(i)} \text{sign}(i-j)\lambda_{j|i}^k) + f_i(y^{\{i\}})]$
    $\lambda_{i|j}^{k+1} = \lambda_{j|i}^k + \text{sign}(i-j)(y^{\{j,k\}}(t) - y^{\{i,k+1\}}(t))$
    $k \leftarrow k+1$
  end while
  for all $i \in \nu$ [in parallel] **do**
    $A_i(t+1) = A_i(t) - \mu A_i(t) F[y^{\{i,k-1\}}(t)]$
  end for
end for

4. EXPERIMENTAL RESULTS

To evaluate the performance of the fusion-center and fully distributed linear BSS algorithms, we implemented them for graphs of 25 nodes with the connectivity pattern shown in Fig. 1.

4.1. Experimental Setup

We considered two human speech signals of 6.687 seconds in length as the original source signals. The signals were sampled at $fs =$ 16KHz. Each node observed a linear instantaneous mixture of the sources through a mixing matrix $A_{25\times 2}$ with elements that were samples from a uniform distribution.

The initial values for the elements of A were obtained by sampling from a normal distribution with zero mean and unit variance. To maximize processing speed, we used only synchronous updating in our experiments. The value for μ for updating the mixing matrix A was set to 0.02. For the ADMM algorithm, the penalty parameter ρ was set to 1 and u_i^1 and z^1 were initialized with zero. We used 100 iterations for source estimation at each time index t for both approaches.

To assess the ability of the proposed de-centralized approach in source signal separation when the number of sources is overestimated, we allocated a source vector of size 4 and considered a matrix A with dimensionality of 25×4.

4.2. Required output power

A major issue in a WSN is the transmission power consumed. According to [22], the required RF output power for a reliable transmission between the nodes of the wireless sensor network is

$$P_{Tx}(d) = \epsilon d^{\alpha} \tag{14}$$

where d is the range of transmission, ϵ is a constant that is determined by the characteristics of the transmitting and receiving

Table 1. The required RF output power in Watt for different target MSEs.

MSE	1×10^{-3}	4×10^{-4}	1×10^{-4}	4×10^{-5}
De-centralized	375ϵ	775ϵ	1575ϵ	2075ϵ
Fusion-center	2052ϵ	4428ϵ	7884ϵ	10152ϵ

antennas and α is the path-loss exponent, which is about 2 for free space. Equation (14) allows us to compare the required power for data transmission when using the aforementioned approaches (fusion-center & de-centralized).

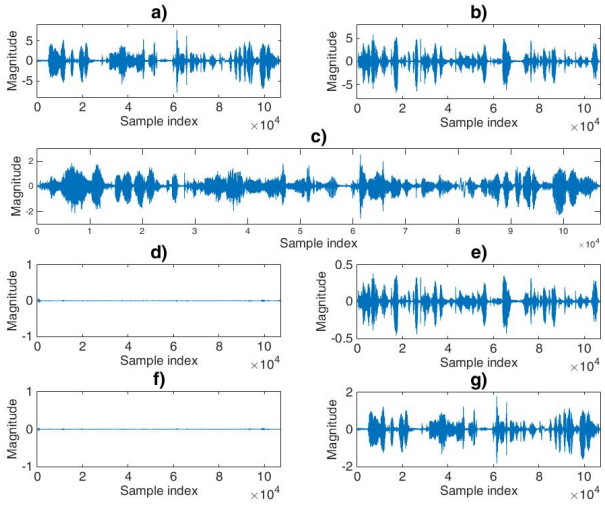

Fig. 2. Visualization of the mixture and separated signals. a) First source signal. b) Second source signal. c) Observed mixture at node 25. d) First separated signal at node 25. e) Second separated signal at node 25. f) Third separated signal at node 25. g) Forth separated signal at node 25.

4.3. Results

Both fusion-center and de-centralized approaches are able to separate the signals effectively. Fig. 2 shows the original signals as well as the reconstructed signals at a sample node for the de-centralized case. As expected, the signals are reconstructed with high accuracy. The methods perform well when the number of sources is unknown. Of the four allotted signals available two show a zero output.

We also investigated the rate of convergence at a given time sample for distributed source estimation. Fig. 3 shows the SNR of the estimated signals at time sample 80000 in the utterances ($t = 5s$). We used the definition $\text{SNR}_i = 10\log_{10}(\frac{||A^+(t)b(t)||_2^2}{||A^+(t)b(t)-y^{\{i,k\}}(t)||_2^2})$. From Figure 3 we see that the convergence rate of the two approaches are similar. The fusion-center approach provides a slightly more consistent convergence behaviour across the nodes, which is natural considering the explicit global averaging step for each iteration.

To check the ability of the proposed approaches in source separation in noisy observations, we contaminated the observed signal at each node with a zero mean Gaussian noise where the energy of the noise was equal across the nodes. Table 2 shows the signal to interference plus noise ratio (SINR) of the separated signals using different methodologies and noise variances. As it can be seen from Table 2, the performance of the distributed approaches is equivalent to the centralized one. The similiarity in performance is expected to increase when higher number of iterations is used for distributed source estimation at each time sample. The latter is the consequence of minimizing the estimation error between distributed approaches and the centralized one in source estimation. The effective performance of the distributed approaches as well as the centralized one in noisy observations is also dominated by the results in Table 2.

Table 2. Shows the SINR of the separated signals in dB at node 25 for different noise variances σ_n^2. S1 and S2 stands for the estimation of the first source signal and second source signal respectively.

	Fusion-center		De-centralized		Centralized	
σ_n^2	S1	S2	S1	S2	S1	S2
0	43.17	15.23	43.28	15.46	44.07	16.27
2×10^{-3}	20.40	13.33	20.55	13.64	21.63	14.66
4×10^{-3}	17.55	10.14	17.74	10.36	18.93	11.27
6×10^{-3}	16.05	8.04	16.15	8.28	17.22	9.31
8×10^{-3}	15.02	7.32	15.14	7.56	16.20	8.60

To evaluate the required radio transmission power, we assumed that the nodes are separated by 1 meter from their neighbours in free space. The required output power for each iteration of the de-centralized and fusion-center approaches were estimated as 25ϵ and 108ϵ watts, respectively. Table 1 shows the required output power for reaching the listed target MSEs at time sample 80000 for the two approaches (with the definition MSE = $\frac{1}{N}\sum_{i=1}^{N}||A^+(t)b(t) - y^{\{i,k\}}(t)||_2^2$). From Table 1 we see that for reaching to a target MSE, the fusion-center approach consumes significantly higher power in comparison with the de-centralized one.

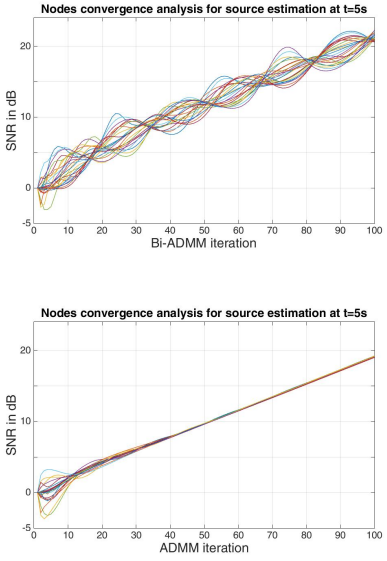

Fig. 3. Comparative performance of fusion-center approach and de-centralized approach in distributed source estimation at time $t = 5s$. Top) De-centralized approach. Bottom) Fusion-center approach.

5. CONCLUSION

We introduced two distributed on-line linear blind source separation algorithms. The de-centralized version of the proposed algorithms benefits from a fully shared computation and can be implemented over any connected graph with any connectivity patterns. Although the fusion-center approach provides a more consistent convergence behaviour for source estimation, it consumes more power in comparison with the de-centralized approach. In terms of scalability, the de-centralized approach also outperforms the fusion-center approach.

6. REFERENCES

[1] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural networks*, vol. 13, no. 4, pp. 411–430, 2000.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[3] E. Robledo-Arnuncio and B. H. Juang, "Blind source separation of acoustic mixtures with distributed microphones," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2007, vol. 3, pp. III 949 – III 952.

[4] J. P. Dmochowski, Z. Liu, and P. A. Chou, "Blind source separation in a distributed microphone meeting environment for improved teleconferencing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2008, pp. 89–92.

[5] S. Gowrishankar, T. G. Basavaraju, D. H. Manjaiah, and S. K. Sarkar, "Issues in wireless sensor networks," in *Proceedings of the World Congress on Engineering*, 2008, vol. 1, pp. 978–988.

[6] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*. ACM, 2004, pp. 20–27.

[7] C. C. Moallemi and B. V. Roy, "Distributed optimization in adaptive networks," in *Advances in Neural Information Processing Systems*, 2003.

[8] A. I. Margaris and K. I. Diamantaras, "A parallel implementation of the natural gradient BSS method using MPI," in *2nd Int. Conference on Experiments/Process/System Modeling/Simulation & Optimization*, 2007.

[9] H. Du, H. Qi, and X. Wang, "A parallel independent component analysis algorithm," in *12th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2006, vol. 1, pp. 151–160.

[10] Y. Hioka and W. B. Kleijn, "Distributed blind source separation with an application to audio signals," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 233–236.

[11] M. Lewicki and T. Sejnowski, "Learning overcomplete representations," *Neural computation*, vol. 12, no. 2, pp. 337–365, 2000.

[12] S. Amari, T. P. Chen, and A. Cichocki, "Nonholonomic orthogonal learning algorithms for blind source separation," *Neural computation*, vol. 12, no. 6, pp. 1463–1484, 2000.

[13] S. Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[14] J. F. Cardoso, "Blind signal separation: statistical principles," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2009–2025, 1998.

[15] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[16] J. Eckstein and W. Yao, "Augmented Lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results," *RUTCOR Research Reports*, vol. 32, 2012.

[17] B. He and X. Yuan, "On the O(1/n) convergence rate of the douglas-rachford alternating direction method," *SIAM Journal on Numerical Analysis*, vol. 50, no. 2, pp. 700–709, 2012.

[18] H. Wang, A. Banerjee, and Z. Q. Luo, "Parallel direction method of multipliers," in *Advances in Neural Information Processing Systems*, 2014, pp. 181–189.

[19] W. Deng, M. J. Lai, Z. Peng, and W. Yin, "Parallel multi-block ADMM with O(1/k) convergence," *arXiv preprint arXiv:1312.3040*, 2013.

[20] E. Wei and A. Ozdaglar, "On the O(1/ k) convergence of asynchronous distributed alternating direction method of multipliers," in *Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2013, pp. 551–554.

[21] G. Zhang and R. Heusdens, "Bi-alternating direction method of multipliers over graphs," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015.

[22] Q. Wang, M. Hempstead, and W. Yang, "A realistic power consumption model for wireless sensor network devices," in *3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON)*. IEEE, 2006, vol. 1, pp. 286–295.