

# Joint Link Learning and Cognitive Radio Sensing

Seung-Jun Kim, Nitin Jain, Georgios B. Giannakis, and Pedro A. Forero

Department of Electrical & Computer Engineering

University of Minnesota

Minneapolis, MN 55455

E-mail: {seungjun, jainx176, georgios, forer002}@umn.edu

**Abstract**—Novel cooperative spectrum sensing algorithms for cognitive radios (CRs) are developed, which can blindly learn the channel gains between CRs and licensed primary users (PUs), while jointly detecting active PU transmitters at each time instant. A dictionary learning approach is taken to decompose the received signal energy samples per CR into linear combinations of channel gains and PU transmit-powers, up to scaling ambiguity. In addition to a batch baseline algorithm, an efficient online implementation that can track slow variation of channel gains is developed, as well as a distributed alternative, which requires only local message passing among neighbors in CR networks. Numerical tests verify the proposed design.

## I. INTRODUCTION

The cognitive radio (CR) paradigm aims at mitigating the increasingly strenuous spectrum scarcity for wireless communication through agile learning and adaptation to the operating environment. The rationale is that although most of the spectrum has been allotted to primary users (PUs) that possess exclusive usage rights, the bands are often significantly underutilized depending on the time and the place of operation. CRs are thus equipped with spectrum sensing capability to identify unused spectral resources—often called “spectrum holes” or “white space”—in the time, frequency, and space domains. Dynamic resource allocation is then performed to make opportunistic use of spectrum without causing harmful interference to the incumbent PU system.

The basic spectrum sensing task is to detect the presence of PU signals in a given frequency band at a given time instant. Such information is useful in spectrum *overlay* scenarios, where CRs are allowed to transmit only on a band completely unoccupied [1]. More sophisticated sensing algorithms can estimate the channel gains, transmit-powers, as well as PU locations. These are valuable in spectrum *underlay* scenarios, where CRs can share bands with PUs as long as the interference inflicted to PUs is kept below a tolerable level.

Spectrum sensing is challenging because prior knowledge on PU signal characteristics may not be available, and PU systems typically do not cooperate for CR sensing. Moreover, fading and shadowing effects make reliable detection difficult. To mitigate these hurdles, multiple CRs can collaborate to perform cooperative spectrum sensing. A typical assumption is that although CRs are spread over a geographical area to benefit from diversity of reception, they are still close to each

other compared to their distances to the PUs such that all CRs observe essentially the same PU activity [2].

More aggressive *spatial* spectrum reuse is possible by departing from such an assumption. RF cartography is a technique that maps out the RF environment in which CR networks operate. Maps of power spectral density provide information as to which regions are less “crowded” in terms of interference, thus yielding more transmission opportunities for CRs [3], [4]. Channel gain maps are instrumental in detecting PU presence as well as calculating the interference caused by CRs to protect PU receivers [5], [6].

In [5] and [6], map tracking algorithms using Kriged Kalman filtering were developed. The algorithms could interpolate channel gains over the entire space from the measurements made at CR positions. To achieve this, however, the CRs must exchange training sequences to obtain sample measurements of channel gains. In the present work, we aim to learn the channel gains *blindly*, while at the same time detecting the activities of PUs, through cooperation of a network of CRs. Online and distributed implementations will permit tracking slowly varying channels via local message passing.

To these ends, a dictionary learning approach is taken. Dictionary learning decomposes a signal into a linear combination of a few atoms from a dictionary, which is jointly learned. Since the seminal work by Olshausen and Field applied this technique to understand human vision [7], it has been used successfully in a number of areas including image denoising, classification, and clustering [8], [9].

The rest of the paper is organized as follows. Sec. II describes the joint link learning and CR sensing problem. Sec. III provides the batch and online solutions. Sec. IV develops a distributed algorithm. Sec. V presents results of numerical tests. Concluding remarks are offered in Sec. VI.

## II. SYSTEM MODEL AND PROBLEM STATEMENT

Consider a CR network consisting of  $M$  nodes that collaborate to detect the presence of active PUs in the deployment area. Assume that there are  $K$  PUs with the  $k$ -th one transmitting at power level  $p_k(t)$  at time  $t$ , for  $k = 1, 2, \dots, K$ . The  $m$ -th CR takes a measurement  $\pi_m(t)$  at time  $t$  of the RF power spectrum in the narrow frequency band of interest. Let  $g_{mk}(t)$  denote the channel gain at time  $t$  between the  $m$ -th CR and the  $k$ -th PU. Due to the lack of explicit support from the PU system on channel estimation, it is often challenging

This work was supported by the QNRF grant NPRP 09-341-2-128.

for CRs to acquire the channels accurately. Thus, knowledge of  $g_{mk}(t)$  is not assumed; rather, it will be learned from the measurements, while jointly detecting active PU transmitters.

The measurement  $\pi_m(t)$  can be modeled as

$$\pi_m(t) = \sum_{k=1}^K g_{mk}(t)p_k(t) + e_m(t), \quad m = 1, 2, \dots, M \quad (1)$$

where  $e_m(t)$  is the measurement error with power  $\sigma^2$ . Define matrix  $\mathbf{G}(t) \in \mathbb{R}_+^{M \times K}$  with  $g_{mk}(t)$  as its  $(m, k)$ -th element. Define  $\boldsymbol{\pi}(t) \triangleq [\pi_1(t), \dots, \pi_M(t)]^T$  and  $\mathbf{p}(t) \triangleq [p_1(t), \dots, p_K(t)]^T$ . Vector  $\mathbf{e}(t)$  is defined likewise. Then, (1) can be compactly written as

$$\boldsymbol{\pi}(t) = \mathbf{G}(t)\mathbf{p}(t) + \mathbf{e}(t). \quad (2)$$

Since oftentimes not all PU transmitters will transmit simultaneously, many of the entries of vector  $\mathbf{p}(t)$  will be zero. This is especially true when the CR network does not have prior knowledge as to how many potential PU transmitters are present. In this case,  $K$  must be set to an upper-bound to the actual number  $K_0$  of the PUs, leading to an even sparser  $\mathbf{p}(t)$ .

Supposing that the PU activity changes fast compared to the measurement interval,  $\mathbf{p}(t)$  will tend to be statistically independent across time. On the other hand, it is assumed that the channel changes slowly relative to the measurement interval. To be specific, suppose for now that the channel gains do not change over  $T$  sampling intervals; that is,  $\mathbf{G}(t) = \mathbf{G}$  for  $t = 1, 2, \dots, T$ . Then, by accumulating the measurements of all CRs over time  $t = 1, 2, \dots, T$  into matrix  $\boldsymbol{\Pi} \triangleq [\boldsymbol{\pi}(1), \boldsymbol{\pi}(2), \dots, \boldsymbol{\pi}(T)] \in \mathbb{R}_+^{M \times T}$ , and likewise defining  $\mathbf{P} \triangleq [\mathbf{p}(1), \dots, \mathbf{p}(T)] \in \mathbb{R}_+^{K \times T}$  as well as  $\mathbf{E}$ , one obtains

$$\boldsymbol{\Pi} = \mathbf{G}\mathbf{P} + \mathbf{E}. \quad (3)$$

Given  $\boldsymbol{\Pi}$ , our goal is to learn the channel matrix  $\mathbf{G}$  and determine the sparse matrix  $\mathbf{P}$ . Since column-wise scaling in  $\mathbf{G}$  and the corresponding inverse scaling applied to the rows of  $\mathbf{P}$  do not alter the product  $\mathbf{G}\mathbf{P}$ , learning only the relative magnitudes of the channels and the transmit-powers is possible, which is a limitation inherent to any blind algorithm.

To accomplish this, a joint dictionary learning and sparse regression formulation is considered. First, define set  $\mathcal{G}$  as

$$\mathcal{G} \triangleq \{\{\mathbf{g}_1, \dots, \mathbf{g}_K\} \geq \mathbf{0} : \|\mathbf{g}_k\|_2 \leq 1 \text{ for } k = 1, \dots, K\} \quad (4)$$

where the nonnegativity constraints are applied element-wise, and  $\{\mathbf{g}_k\}$  are column vectors. Then, the optimization problem is posed as

$$\min_{\mathbf{G} \in \mathcal{G}, \mathbf{P} \geq \mathbf{0}} \frac{1}{2} \|\boldsymbol{\Pi} - \mathbf{G}\mathbf{P}\|_F^2 + \lambda \|\mathbf{P}\|_1 \quad (5)$$

where the constraint  $\mathbf{G} \in \mathcal{G}$  prevents the columns of  $\mathbf{G}$  growing arbitrarily large, in addition to enforcing nonnegativity, and  $\|\mathbf{P}\|_1$  is the sum of the  $\ell_1$ -norms of the columns of  $\mathbf{P}$ , which is a penalty term promoting sparsity in  $\mathbf{P}$ . Since each entry in  $\mathbf{P}$  is nonnegative,  $\|\mathbf{P}\|_1$  is simply equal to the sum of all the entries in  $\mathbf{P}$ . Parameter  $\lambda$  tunes the sparsity level of  $\mathbf{P}$ .

Input: $\boldsymbol{\Pi}$ , initial dictionary $\mathbf{G}_0$ , and $\lambda$
Output: $\mathbf{G}$ and $\mathbf{P}$
1: Set $\mathbf{G} \triangleq [\mathbf{g}_1, \dots, \mathbf{g}_K] = \mathbf{G}_0$
2: Repeat
3:     Perform sparse regression with fixed $\mathbf{G}$ .
4:     For $t = 1, 2, \dots, T$
5: $\mathbf{p}(t) = \arg \min_{\mathbf{p} \geq \mathbf{0}} \frac{1}{2} \ \boldsymbol{\pi}(t) - \mathbf{G}\mathbf{p}\ _2^2 + \lambda \ \mathbf{p}\ _1$
6:     Next $t$
7:     Perform dictionary update with fixed $\mathbf{P}$ .
8:     Compute $\mathbf{A} = \mathbf{P}\mathbf{P}^T$ and $\mathbf{B} = \boldsymbol{\Pi}\mathbf{P}^T$
9:     Repeat
10:         For $k = 1, 2, \dots, K$
11: $\tilde{\mathbf{g}}_k = \frac{1}{A_{kk}} (\mathbf{b}_k - \mathbf{G}\mathbf{a}_k) + \mathbf{g}_k$
12: $\mathbf{g}_k = \frac{[\tilde{\mathbf{g}}_k]^+}{\max\{\ \tilde{\mathbf{g}}_k\ _2, 1\}}$
13:         Next $k$
14:     until convergence
15: until convergence

TABLE I  
BATCH ALGORITHM.

Instead of the batch setup considered in (5), an online formulation may be favored when the channel gains are continuously time-varying, albeit still slowly compared to the measurement interval. The online solver will then be able to track the channels. Moreover, the computation may be done recursively as new measurements are obtained over time, thus considerably reducing the computational burden. For this, consider solving at each time  $t$

$$\min_{\substack{\mathbf{G} \in \mathcal{G}, \\ \mathbf{p}(s) \geq \mathbf{0}}} \sum_{s=1}^t \beta^{t-s} \left( \frac{1}{2} \|\boldsymbol{\pi}(s) - \mathbf{G}\mathbf{p}(s)\|_2^2 + \lambda \|\mathbf{p}(s)\|_1 \right) \quad (6)$$

where the weights  $\beta^{t-s}$  with  $\beta \in (0, 1]$  are used to gradually forget the past measurements while weighting the recent measurements more heavily.

### III. JOINT LINK LEARNING AND SENSING ALGORITHMS

Problems (5) and (6) are nonconvex. Therefore, attaining the global optimum may be difficult. However, when  $\mathbf{G}$  is fixed, optimizing over  $\mathbf{P}$  involves convex optimization. Likewise, when  $\mathbf{P}$  is fixed, the problems for  $\mathbf{G}$  become convex. Therefore, alternating minimization is employed to obtain locally optimal solutions. Next, both batch and online algorithms are developed.

#### A. Batch Algorithm

Various algorithms have been developed for the joint dictionary learning and sparse coding problem (5). A small but important difference of our setup from the most common ones is the nonnegativity constraints imposed on the entries of  $\mathbf{G}$  and  $\mathbf{P}$ . In [10], the  $\ell_2$ -constrained dictionary learning problem was tackled in the dual domain. The  $K$ -SVD algorithm was developed in [11], which generalized the  $K$ -means clustering algorithm to the dictionary learning formulation. However, it is not straightforward to extend these algorithms to account for the nonnegativity constraints.

In this paper, the block coordinate descent method also used in [12] is employed for learning  $\mathbf{G}$  given  $\mathbf{P}$ . In Table I, the batch algorithm is presented. Lines 3–6 yield the sparse vector

Input: $\boldsymbol{\pi}(t)$ for $t = 1, 2, \dots$ , initial dictionary $\mathbf{G}(0)$ , and $\lambda$
Output: $\mathbf{G}(t)$ and $\mathbf{p}(t)$ for $t = 1, 2, \dots$
1: Set $\mathbf{A}(0) = \mathbf{B}(0) = \mathbf{0}$ .
2: For $t = 1, 2, \dots$
3: Perform sparse regression.
4: $\mathbf{p}(t) = \arg \min_{\mathbf{p} \geq 0} \frac{1}{2} \ \boldsymbol{\pi}(t) - \mathbf{G}(t-1)\mathbf{p}\ _2^2 + \lambda \ \mathbf{p}\ _1$
5: Perform dictionary update.
6: Compute:
7: $\mathbf{A}(t) = \beta \mathbf{A}(t-1) + \mathbf{p}(t)\mathbf{p}(t)^T$
8: $\mathbf{B}(t) = \beta \mathbf{B}(t-1) + \boldsymbol{\pi}(t)\mathbf{p}(t)^T$
9: Set $\mathbf{G} \triangleq [\mathbf{g}_1, \dots, \mathbf{g}_K] = \mathbf{G}(t-1)$ (warm start)
10: Repeat
11: For $k = 1, 2, \dots, K$
12: $\tilde{\mathbf{g}}_k = \frac{1}{A_{kk}(t)} [\mathbf{b}_k(t) - \mathbf{G}\mathbf{a}_k(t)] + \mathbf{g}_k$
13: $\mathbf{g}_k = \frac{[\tilde{\mathbf{g}}_k]^+}{\max\{\ \tilde{\mathbf{g}}_k\ _2, 1\}}$
14: Next $k$
15: until convergence
16: (Optional: replace all-zero columns in $\mathbf{G}$ to random vectors.)
17: Set $\mathbf{G}(t) = \mathbf{G}$ .
18: Next $t$

TABLE II  
ONLINE ALGORITHM.

$\mathbf{p}(t)$  for each  $t = 1, 2, \dots, T$  given the dictionary from the previous iteration. This can be solved, e.g., using the LARS algorithm [13]. Lines 7–14 find the  $\mathbf{G}$  that minimizes the quadratic cost  $\|\boldsymbol{\Pi} - \mathbf{G}\mathbf{P}\|_F^2$  subject to  $\mathbf{G} \in \mathcal{G}$ , given  $\mathbf{P}$ . Based on the block coordinate descent method [14, Sec. 2.7], the algorithm sequentially updates the individual columns of  $\mathbf{G}$  with the other ones fixed. Line 11 solves the unconstrained minimization with respect to the  $k$ -th column of  $\mathbf{G}$ , followed by the projection onto the nonnegative portion of the unit  $\ell_2$ -ball in line 12. Here,  $[\cdot]^+$  represents  $\max\{0, \cdot\}$ . Also,  $A_{kk}$  denotes the  $k$ -th diagonal element of matrix  $\mathbf{A} \triangleq \mathbf{P}\mathbf{P}^T$ , and  $\mathbf{a}_k$  and  $\mathbf{b}_k$  represent the  $k$ -th columns of  $\mathbf{A}$  and  $\mathbf{B} \triangleq \boldsymbol{\Pi}\mathbf{P}^T$ , respectively.

### B. Online Algorithm

For online link learning and spectrum sensing, (6) is solved. With a slight modification to account for the forgetting factor  $\beta$  and the nonnegativity constraint, the algorithm in [12] can be adapted as summarized in Table II.

The key difference of the online algorithm from the batch one is that the sparse coding problem is solved only for the “current” power vector  $\mathbf{p}(t)$ ; the past solutions  $\mathbf{p}(t-1), \dots, \mathbf{p}(1)$  are unchanged. Still, it was shown in [12] that under mild conditions,  $\mathbf{G}(t)$  so obtained converges to the same  $\mathbf{G}$  that would have been obtained by updating the entire  $\mathbf{p}(1), \dots, \mathbf{p}(t)$ , as  $t \rightarrow \infty$ . Since the past power vectors do not change, matrices  $\mathbf{A}$  and  $\mathbf{B}$  can be updated recursively as shown in lines 6–8 of Table II. Sometimes some columns of  $\mathbf{G}$  become zeros, degrading performance. Line 16 replaces the degenerate columns with nonzero random columns.

## IV. DISTRIBUTED ONLINE ALGORITHM

The algorithms put forth in the preceding section require a fusion center that collects the measurements from CR nodes to perform dictionary learning in a centralized fashion. In practice, a distributed implementation may be desired, e.g., to

Input: $\{\pi_m(t)\}_{m=1}^M$ for $t = 1, 2, \dots$ , initial dictionary $\{\tilde{\mathbf{g}}_m(0)\}_{m=1}^M$ , and $\lambda$
Output: $\{\tilde{\mathbf{g}}_m(t)\}_{m=1}^M$ and $\mathbf{p}_1(t) = \mathbf{p}_2(t) = \dots = \mathbf{p}_M(t)$ for $t = 1, 2, \dots$
1: Set $\mathbf{A}_m(0) = \mathbf{0}$ and $\mathbf{b}_m(0) = \mathbf{0}$ for all $m$ .
2: For $t = 1, 2, \dots$
3: Perform distributed sparse regression.
4: Initialize $\zeta_m^{(0)}, \kappa_m^{(0)}, \mathbf{p}_m^{(1)}$ and $\mathbf{q}_m^{(1)}$ arbitrarily for all $m$ (e.g., warm start)
5: For $i = 1, 2, \dots, \text{MAX\_ITER}$
6: Perform updates in (8)–(11) at each CR $m \in \{1, 2, \dots, M\}$
7: Next $i$
8: Perform distributed dictionary update.
9: At each CR $m \in \{1, 2, \dots, M\}$ in parallel
10: Compute:
11: $\mathbf{A}_m(t) = \beta \mathbf{A}_m(t-1) + \eta \mathbf{I} + \mathbf{p}_m(t)\mathbf{p}_m(t)^T$
12: $\mathbf{b}_m(t) = \beta \mathbf{b}_m(t-1) + \mathbf{p}_m(t)\pi_m(t)$
13: Set $\tilde{\mathbf{g}}_m \triangleq [g_{m1}, \dots, g_{mK}] = \tilde{\mathbf{g}}_m(t-1)$ (warm start).
14: Repeat
15: For $k = 1, 2, \dots, K$
16: $g_{mk} = \left[ \frac{1}{A_{m,kk}(t)} (\mathbf{b}_{mk}(t) - \tilde{\mathbf{g}}_m \mathbf{a}_k(t)) + g_{mk} \right]^+$
17: Next $k$
18: until convergence
19: Set $\tilde{\mathbf{g}}_m(t) = \tilde{\mathbf{g}}_m$
20: (Optional: if the $k$ -th entry of $\mathbf{p}_m(t)$ has been zero for consecutive MAX_UNUSED times, replace it with a random number, for $k = 1, \dots, K$ )
21: Next $t$

TABLE III  
DISTRIBUTED ONLINE ALGORITHM.

avoid the single point of failure, ensure scalability of the algorithm, and reduce the feedback overhead. Thus, consensus-based optimization over a multi-hop network is employed to obtain a *distributed* online dictionary learning algorithm [15].

One hurdle in deriving a distributed algorithm to solve (6) is the constraint  $\mathbf{G} \in \mathcal{G}$ . Since this constraint couples all CRs, a naive implementation requires centralized processing. To bypass this issue, the following alternative formulation is considered; see also [16]

$$\min_{\substack{\mathbf{G} \geq 0 \\ \mathbf{p}(t) \geq 0}} \sum_{s=1}^t \beta^{t-s} \left[ \frac{1}{2} (\|\boldsymbol{\pi}(s) - \mathbf{G}\mathbf{p}(s)\|_2^2 + \eta \|\mathbf{G}\|_F^2) + \lambda \|\mathbf{p}(s)\|_1 \right] \quad (7)$$

where the extra  $\ell_2$  regularization term on  $\mathbf{G}$  prevents the entries of  $\mathbf{G}$  from growing without bound.

Similar to the centralized algorithms, the distributed one works by alternating two steps: distributed sparse regression and distributed dictionary update. The goal of the distributed sparse regression step is to solve the optimization problem in line 4 of Table II through in-network processing. That is, it is desired that each CR  $m \in \{1, 2, \dots, M\}$  takes its own measurement  $\pi_m(t)$  at time  $t$ , and arrives at global consensus on the optimal  $\mathbf{p}(t)$  in a distributed manner, using only the messages exchanged with its one-hop neighbors  $\mathcal{N}_m \subset \{1, 2, \dots, M\}$ . This problem has been tackled in [17] and [5]. In particular, the algorithm in [17] allows parallel update of the individual entries in  $\mathbf{p}(t)$ . We only make minor modifications to take care of nonnegativity of  $\mathbf{p}(t)$ .

It is emphasized that only the  $m$ -th row of  $\mathbf{G}(t-1)$  is

necessary to perform the updates at each CR  $m \in \{1, \dots, M\}$ . Let  $\bar{\mathbf{g}}_m(t-1)$  denote the  $m$ -th row of  $\mathbf{G}(t-1)$ . Then, the following iterative update is performed at CR  $m$ , with  $i$  denoting the iteration index:

$$\zeta_m^{(i)} = \zeta_m^{(i-1)} + \rho \sum_{m' \in \mathcal{N}_m} (\mathbf{p}_m^{(i)} - \mathbf{p}_{m'}^{(i)}) \quad (8)$$

$$\kappa_m^{(i)} = \kappa_m^{(i-1)} + \rho (\mathbf{p}_m^{(i)} - \mathbf{q}_m^{(i)}) \quad (9)$$

$$\mathbf{p}_m^{(i+1)} = \frac{1}{\rho(2|\mathcal{N}_m|+1)} \left[ \rho \mathbf{q}_m^{(i)} - \zeta_m^{(i)} - \kappa_m^{(i)} + \rho \sum_{m' \in \mathcal{N}_m} (\mathbf{p}_m^{(i)} + \mathbf{p}_{m'}^{(i)}) - \frac{\lambda}{M} \right]^+ \quad (10)$$

$$\mathbf{q}_m^{(i+1)} = \rho^{-1} \left[ \mathbf{I} - \frac{\bar{\mathbf{g}}_m(t-1)^T \bar{\mathbf{g}}_m(t-1)}{\rho + \|\bar{\mathbf{g}}_m(t-1)\|_2^2} \right] \cdot (\bar{\mathbf{g}}_m(t-1)^T \pi_m(t) + \rho \mathbf{p}_m^{(i+1)} + \kappa_m^{(i)}) \quad (11)$$

Here,  $|\cdot|$  is the cardinality of the set,  $\rho > 0$  a step size parameter, and  $\zeta_m$ ,  $\kappa_m$  and  $\mathbf{q}_m$  are auxiliary variables. It can be seen that CR  $m$  just needs to collect locally from its neighbors  $m' \in \mathcal{N}_m$  the current estimate of  $\mathbf{p}_{m'}$  at each iteration. From the theory of alternating direction method of multipliers (ADMoM) on which the algorithm is based, it can be shown that the algorithm converges globally as  $i$  tends to infinity, to the consensus estimate  $\mathbf{p}_1^{(i)} = \mathbf{p}_2^{(i)} = \dots = \mathbf{p}_M^{(i)}$ , which coincides with the solution  $\mathbf{p}(t)$  to the problem in line 4 of Table II, provided that the network is connected; i.e., there exist (multi-hop) paths from any node to any other node.

Based on the shared  $\{\mathbf{p}(s)\}_{s=1}^t$  accumulated up to time  $t$ , the dictionary  $\mathbf{G}$  can be updated in a distributed manner. This is straightforward since the cost in (7) as well as the constraint  $\mathbf{G} \geq \mathbf{0}$  are separable in the individual rows of  $\mathbf{G}$ . Thus, each CR  $m$  needs to update  $\bar{\mathbf{g}}_m$  by solving a nonnegativity-constrained least-squares problem. Obviously, this can be done in parallel. For simplicity, the coordinate descent method is used again in the proposed algorithm. The overall algorithm is presented in Table III, where  $A_{m,kk}(t)$  denotes the  $(k,k)$ -th entry of matrix  $\mathbf{A}_m(t)$ , and  $b_{mk}(t)$  the  $k$ -th entry of vector  $\mathbf{b}_m(t)$ . Line 20 replaces degenerate columns in  $\mathbf{G}$  with random columns in a distributed manner.

## V. NUMERICAL TESTS

To verify the proposed algorithms, numerical tests are performed. A network of  $M = 10$  CR nodes is used to detect  $K_0 = 3$  PUs. The locations of the CRs and the PUs are depicted in Fig. 1 using circles and triangles, respectively, while single-hop connectivity is represented by the edges of the graph. The number of atoms in the dictionary is set to  $K = 5$ . The channels  $g_{mk}$  are given by  $g_{mk} = (d_{mk}/d_0)^{-\gamma} |h_{mk}|^2$  for  $m = 1, \dots, M$  and  $k = 1, \dots, K_0$ , where  $d_{mk}$  is the distance between CR  $m$  and PU  $k$ ,  $d_0$  is the reference distance set to 0.1,  $\gamma$  is the path loss exponent set to 2.5, and  $h_{mk}$  are Rayleigh fading coefficients following an autoregressive model

$$h_{mk}(t+1) = \alpha h_{mk}(t) + \sqrt{1 - \alpha^2} z_{mk}(t) \quad (12)$$

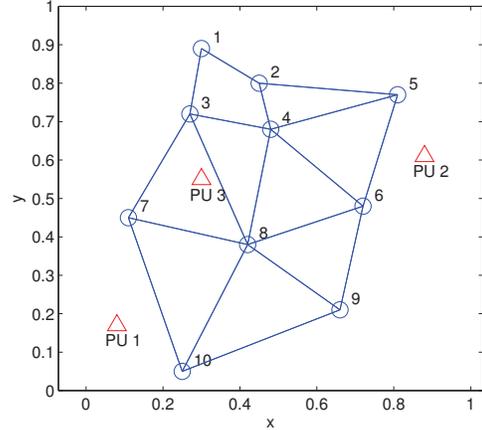


Fig. 1. CR network topology.

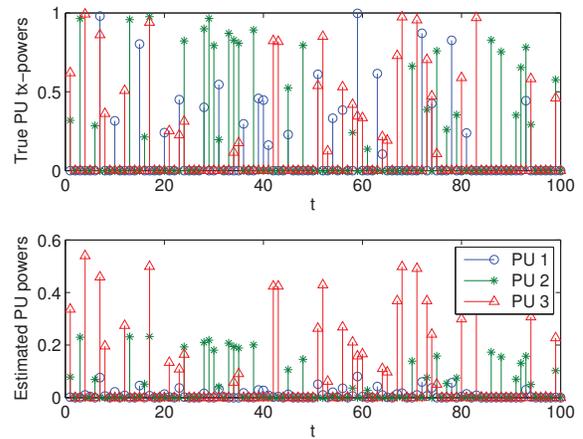


Fig. 2. True and estimated PU transmit-powers.

with  $h_{mk}$  and  $z_{mk}$  drawn independently from the standard complex Gaussian distribution. When a time-varying channel is generated,  $\alpha = 0.9999$  is used, while for fixed channels,  $\alpha = 1$ . The PUs are active with probability 0.3, and upon being active, their transmission powers are drawn independently from the uniform distribution with support  $[0.1, 1]$ .

Fig. 2 shows the true PU transmit-powers (top panel) and the estimated PU powers (bottom panel). The channels were time-varying and the (centralized) online algorithm with  $\beta = 0.95$  was employed. To identify the PUs detected, correlation coefficients between the true and the estimated channels were computed and thresholded at 0.9. When more than one atoms of the dictionary had correlation higher than the threshold, the corresponding entries in  $\mathbf{p}(t)$  were summed. It can be seen from the figure that the relative magnitudes of PU powers have been correctly estimated. Note that since the channels are normalized, the distance-based path loss effect is captured in the estimated PU powers; cf. Fig. 1. Moreover, the network sensing algorithm can detect the correct set of active PUs from the mixture observations, by checking whether the estimated PU powers are positive. Fig. 3 depicts the corresponding receiver operating characteristics (ROCs) for detecting the individual PUs.

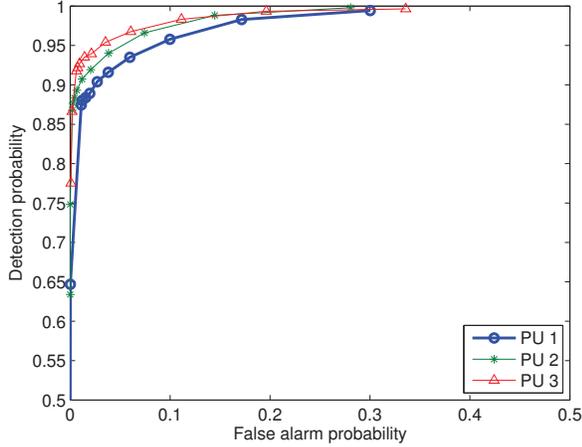


Fig. 3. ROC curves.

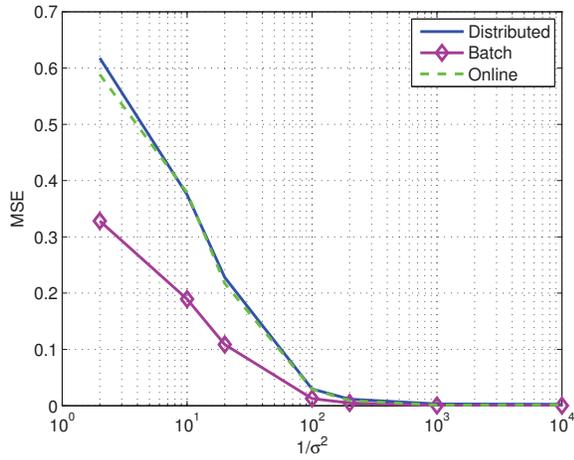


Fig. 4. MSE vs. inverse of noise variance.

To assess the quality of channel estimation, the normalized mean-square error (MSE) performance of the batch, online and distributed algorithms as the noise power is varied is presented in Fig. 4 for the case where the channels are fixed. It can be seen that the batch algorithm outperforms the online versions especially at low SNR. On the other hand, the performance of the centralized and the distributed online algorithms is indistinguishable. Similar observation can be made from the MSE-versus-time curves of Fig. 5 for time-varying channels, where the noise power is  $10^{-3}$ .

## VI. CONCLUSIONS

Cooperative spectrum sensing algorithms have been proposed that can detect the active set of PUs at each time instant while jointly acquiring the channels between CRs and PUs. Employing a dictionary learning approach, the algorithms factorized the received energy sample matrix into a dictionary with nonnegative entries that represented normalized channel gains, and a sparse nonnegative matrix signifying PU transmit-powers. Batch, online, as well as distributed online algorithms were developed. The numerical tests demonstrated that the

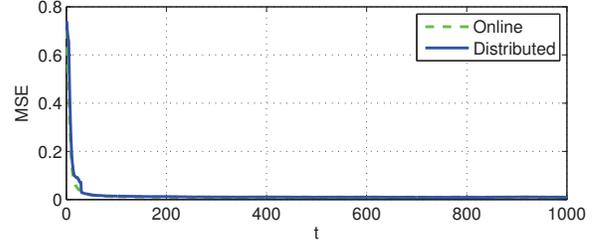


Fig. 5. MSE vs. time.

algorithms can estimate individual PU powers and track slowly varying channels, up to scaling ambiguity. Resolving scaling ambiguity through additional constraints as well as online tuning of regularization parameters are left for future work.

## REFERENCES

- [1] Q. Zhao and B. M. Sadler, "A survey of dynamic spectrum access," *IEEE Sig. Proc. Mag.*, vol. 24, no. 3, pp. 79–89, May 2007.
- [2] J. Unnikrishnan and V. V. Veeravalli, "Cooperative sensing for primary detection in cognitive radio," *IEEE J. Sel. Topics Sig. Proc.*, vol. 2, no. 1, pp. 18–27, Feb. 2008.
- [3] A. Alaya-Feki, S. Ben Jemaa, B. Sayrac, P. Houze, and E. Moulines, "Informed spectrum usage in cognitive radio networks: interference cartography," in *Proc. of IEEE 19th Intl. Symp. Personal, Indoor and Mobile Radio Commun.*, Cannes, France, Sep. 2008, pp. 1–5.
- [4] J. A. Bazerque, G. Mateos, and G. B. Giannakis, "Group-Lasso on splines for spectrum cartography," *IEEE Trans. Sig. Proc.*, vol. 59, no. 10, pp. 4648–4663, Oct. 2011.
- [5] S.-J. Kim, E. Dall'Anese, and G. B. Giannakis, "Cooperative spectrum sensing for cognitive radios using Kriged Kalman filtering," *IEEE J. Sel. Topics Sig. Proc.*, vol. 5, no. 1, pp. 24–36, Feb. 2011.
- [6] E. Dall'Anese, S.-J. Kim, and G. B. Giannakis, "Channel gain map tracking via distributed Kriging," *IEEE Trans. Veh. Technol.*, vol. 60, no. 3, pp. 1205–1211, Mar. 2011.
- [7] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision Research*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.
- [8] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Proc.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.
- [9] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: transfer learning from unlabeled data," in *Proc. of the 24th Intl. Conf. on Machine Learning*, Corvallis, OR, Jun. 2007, pp. 759–766.
- [10] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds. Cambridge, MA: MIT Press, 2007, pp. 801–808.
- [11] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Sig. Proc.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
- [12] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *J. Machine Learning Res.*, vol. 11, pp. 19–60, Mar. 2010.
- [13] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Stat.*, vol. 32, no. 2, pp. 407–499, 2004.
- [14] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.
- [15] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links—Part I: Distributed estimation of deterministic signals," *IEEE Trans. Sig. Proc.*, vol. 1, no. 56, pp. 350–364, Jan. 2008.
- [16] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for spectral data analysis," *Linear Algebra and its Appl.*, vol. 416, no. 1, pp. 29–47, Jul. 2006.
- [17] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Sig. Proc.*, vol. 58, no. 10, pp. 5262–5276, Oct. 2010.