# Binary Inference for Primary User Separation in Cognitive Radio Networks

Huy Nguyen†, Guanbo Zheng‡, Zhu Han‡, and Rong Zheng†
†Department of Computer Science
‡Department of Electrical and Computer Engineering
University of Houston, Houston, TX 77204
E-mail: {*hanguyen5, gzheng3, zhan2, rzheng*}*@uh.edu*

*Abstract*—**Spectrum sensing receives much attention recently in the cognitive radio (CR) network research, i.e., secondary users (SUs) constantly monitor channel condition to detect the presence of the primary users (PUs). In this paper, we go beyond spectrum sensing and introduce the *PU separation problem*, which concerns with the issues of distinguishing and characterizing PUs in the context of collaborative spectrum sensing and monitor selection. The observations of monitors are modeled as boolean OR mixtures of underlying binary sources for PUs. We first justify the use of the binary OR mixture model as opposed to the traditional linear mixture model through simulation studies. Then we devise a novel binary inference algorithm for PU separation. Not only PU-SU relationship are revealed, but PUs' transmission statistics and activities at each time slot can also be inferred. Simulation results show that without any prior knowledge regarding PUs' activities, the algorithm achieves high inference accuracy even in the presence of noisy measurements.**

## I. INTRODUCTION

With tremendous growth in wireless services, the demand for radio spectrum has significantly increased. However, spectrum resources are scarce and most of them have been already licensed to existing operators. Recent studies have shown that despite claims of spectral scarcity, the actual licensed spectrum remains unoccupied for long periods of time [**?**]. Thus, cognitive radio (CR) systems have been proposed [1], [2], [3] in order to efficiently exploit these spectral holes, in which licensed primary users (PUs) are not present. CRs or secondary users (SUs) are wireless devices that can intelligently monitor and adapt to their environment, hence, they are able to share the spectrum with the licensed PUs, operating when the PUs are idle.

One key challenge in CR systems is spectrum sensing, i.e., SUs attempt to learn the environment and determine the presence and characteristics of PUs. Spectrum sensing can be done at SUs individually or cooperatively [4], [5], with or without the assistance of infrastructure supports such as dedicated monitor nodes and cognitive pilot channel (CPC) [6], [7], [8], [9]. Energy detection is one of the most commonly used method for spectrum sensing, where the detector computes the energy of the received signals and compares it to a certain

threshold value to decide whether the PU signal is present or not. It has the advantage of short detection time but suffers from low accuracy compared to feature-based approaches such as cyclostationary detection [2], [3]. From the prospective of a CR system, it is often insufficient to detect PU activities in a single SU's vicinity ("is there any PU near me?"). Rather, it is important to determine the identity of PUs ("who is there?") as well as the distribution of PUs in the field ("where are they?"). We call these issues the *PU separation problem*.

To motive the need for PU separation, let us consider the following scenarios:

- Multiple SUs cooperatively infer the activities of PUs, some of which may be observable to only a subset of SUs. In this case, the SUs need to identify the PU-SU adjacency relationships. Blindly assuming all PUs are observable to all SUs will lead to inferior detection results.
- Dedicated monitors are employed for spectrum sensing. There exists redundancy in monitors' observations due to common PUs across multiple monitors. Such redundancy can be reduced by judiciously selecting a subset of monitors to report their spectrum sensing results. Furthermore, some monitors can be put to low-power modes for energy conservation.

Clearly, PU separation is a more challenging problem compared to node-level PU detection. The conventional wisdom suggests that sophisticated methods such as feature-based detection are necessary. On the contrary, we find that through cooperation among monitors or SUs, not only accuracy of energy detection can be improved as been demonstrated in several existing work [10], [4], [5], but also PUs can be identified using solely binary information (due to thresholding in energy detection). The key to this surprising result is a binary inference framework that models the observations of SUs and monitors as boolean OR mixtures of underlying binary latency sources for PUs. It allows us to exploit the correlation structure among distributed binary observations. We develop an iterative algorithm, called Binary Independent Component Analysis (bICA), to determine the underlying latent sources (i.e., PUs) and their active probabilities. In bICA, no prior information regarding the PUs' activities or observation noise is assumed. Given $m$ monitors or SUs, up

to $2^m - 1$ PUs can be inferred resulting in great efficiency. Evaluation results show effectiveness of bICA under practical settings.

**Contributions:** In this paper, we make the following contributions toward the design of a binary inference framework for PU separation in cognitive radio networks:

- We introduce the PU separation problem with cooperative SU inference model and discuss its importance on CR systems.
- We provide a stochastic analysis on the difference between linear and binary PU energy detection models. Results from the study imply that using just binary observations from SUs has comparable accuracy with using a linear model, while incurring much less overhead.
- We apply bICA to solve the PU separation problem without any assumption on the noise model or prior knowledge on the PU activities. We furthermore consider the inverse problem of inferring the detailed PUs' activities given the SUs' observations and the inferred model.

The rest of the paper is organized as follows. In Section II, the observation model is introduced. A comparison between the linear and binary energy model and brief overview of related work are also presented. In Section III, we present the bICA algorithm to determine the statistics of PU activities and the inference algorithm to decide which set of PUs are active. Formulation and solution to the inverse problem under noisy measurements are presented in Section IV. Evaluation results are detailed in Section V followed by an overview of related work is provided in Section VI followed by conclusions in Section VII.

## II. MODEL AND PRELIMINARY

Consider a slotted system in which the transmission activities of $n$ PUs are modeled as a set of independent binary variables $\mathbf{y}$ with active probabilities $\mathcal{P}(\mathbf{y})$. The binary observations due to energy detection at the $m$ monitor nodes (for the remaining of the paper, we do not distinguish monitor nodes and SUs) are modeled as an $m$-dimension binary vector $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T$ with joint distribution $\mathcal{P}(\mathbf{x})$. It is assumed that presence of any active PU surrounding of a monitor leads to positive detection. An unknown binary mixing matrix $\boldsymbol{G}_{m \times n}$ is used to represent the relationship between the observable variables in $\mathbf{x}$ and the latent binary variables in $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ as follows:

$$x_i = \bigvee_{j=1}^{n} (g_{ij} \wedge y_j), \; i = 1, \ldots, m, \tag{1}$$

where $\wedge$ is Boolean *AND*, $\vee$ is Boolean *OR*, and $g_{ij}$ is the entry on the $i$th row and the $j$th column of $\boldsymbol{G}$. For ease of presentation, we introduce a short-hand notation as

$$\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}. \tag{2}$$

In essence, $g_{ij}$ encodes whether monitor $i$ can detect the transmission of PU $j$. For a monitor $i$, the energy detection returns 1 when the monitor can detect one or more active PUs.
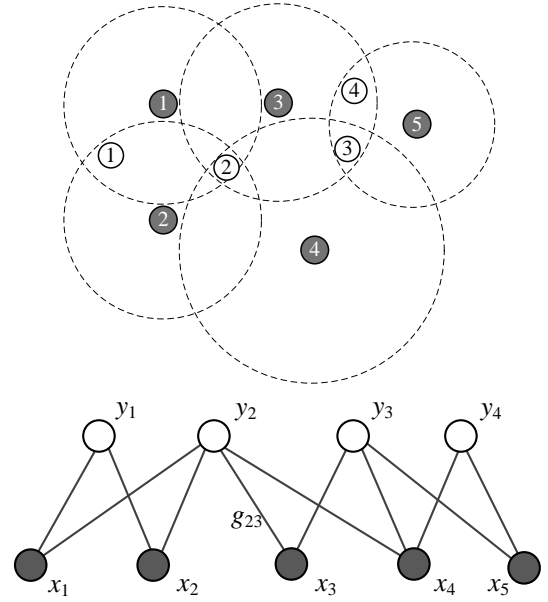


Fig. 1: A sample network scenario with number of SUs $m = 5$, number of PUs $n = 4$ and its bipartite graph transformation. White circles represent PUs, black circles represent SUs and dashed lines illustrate SUs' monitor range.

$\boldsymbol{G}$ can be seen as the adjacency matrix of an undirected bipartite graph $G = (U, V, E)$, where $U = \{x_1, x_2, \ldots, x_m\}$ and $V = \{y_1, y_2, \ldots, y_n\}$. An edge $e = (x_i, y_j)$ exists if $g_{ij} = 1$. Illustration of a sample network scenario and its bipartite graph is presented in Figure 1.

Consider an $m \times T$ matrix $\boldsymbol{X}$, which is the collection of $T$ realizations of vector $\mathbf{x}$. The goal of bICA is to determine the distribution of the latent independent random variables $\mathbf{y}$ and the binary mixing matrix $\boldsymbol{G}$ from $\boldsymbol{X}$, such that $\boldsymbol{X}$ can be decomposed into the mixing of realizations of $\mathbf{y}$. From $\boldsymbol{G}$ and $\mathbf{y}$, we can identify the PUs and additionally infer PUs' activities at different time slots. Note that in (1), measurement noise is not explicitly modeled, rather, is treated as independent sources.

### A. Why Binary Inference?

In this section, we motivate the use of a binary inference framework by considering an alternative linear mixing model. In the linear model, at each time slot, the received signal power at each monitor can be modeled as a linear combination of the transmitted signal power from active PUs. More specifically, Let $\boldsymbol{v} = [v_1, v_2, \ldots, v_m]^T$, $\mathbf{z} = [z_1, z_2, \ldots, z_n]^T$, and $\mathbf{n} = [n_1, n_2, \ldots, n_m]^T$ be the random vectors corresponding to the received, transmitted signal power and the Gaussian noise respectively, and $\boldsymbol{H}$ is the $m \times n$ unknown channel gain matrix. Both large-scale path loss with propagation loss factor $\alpha$ and small-scale fading following the Rayleigh distribution are considered in this model. The received signal power is a linear mixture of the transmitted signal power and the noise:

$$\boldsymbol{v} = \boldsymbol{H}\mathbf{z} + \mathbf{n}. \tag{3}$$

$n_i$ is a random variable with mean $\mu_i$. If $\boldsymbol{v}$ can be observed directly, classical linear independent component analy-
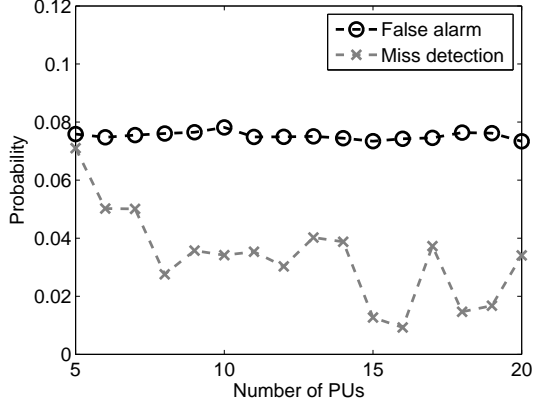
Fig. 2: False alarm and miss detection probability. Threshold $\tau$ is set at 5dB.

sis (ICA) [11] can be applied to determine $\boldsymbol{H}$ and $\mathbf{z}$. However, this method suffers from three problems. First, $z_i$'s have to be non-Gaussian to be recoverable. Second, the channel gain matrix $\boldsymbol{H}$ tends to vary over time. Lastly, communicating the realizations of $\mathbf{z}$ to a centralized server for inference incurs higher overhead compared to its binary counter part $\mathbf{x}$. Furthermore, when only the binary quantized values of $\boldsymbol{v}$ are observable (i.e., $\mathbf{x}$), ICA is not longer suitable due to the non-linearity of the quantization function.

Next, we compare accuracy of the binary OR mixture model against a quantized version of the above linear model, namely, $\mathbf{x}' = U(\boldsymbol{v} - \tau)$, where $U(\cdot)$ is a step function defined as $U(r) = 1$ if $r > 0$, $U(r) = 0$ otherwise, and $\tau$ is a pre-set threshold. We are interested to see the degree of "information loss" due to the OR approximation. In the simulation, 10 monitors and $n$ PUs are deployed in a 500x500 square meter area with $n$ varying from 5 to 20. Locations of PUs are chosen arbitrarily with the restriction that no two PUs can be observed by the same set of monitors. The PUs' transmit power levels are fixed at 20mW, the noise floor is -95dbm, and the propagation loss factor is 3. The SNR detection threshold for the monitors is set to be 5dB (above the noise floor). The value is chosen so that the false alarm probability (PU's are reported while none exists) is less than 10%. Elements of the binary mixing matrix $\boldsymbol{G}$ are either 1 or 0, depending on the received signal for one respective PU only. In other words, $g_{ij} = 1$ if the $i$th SU can detect transmissions from the $j$th PU. The PUs' activities are modeled as a two-stage Markov chain with transition probabilities uniformly distributed over (0,1). We run the simulation for $T = 5,000$ slots and obtain the observations $\boldsymbol{X}$ and $\boldsymbol{X}_l$ for the binary OR and linear mixing model, respectively. Figure 2 shows the false alarm and miss detection probability in the binary OR model using the results from the linear mixing model as the ground truth. From Figure 2, we see the two models have very close performance. We also experiment the case in which the initial phases of PU's transmitted signal vary in $[0, 2\pi]$, and have similar observations.

To this end, we conclude that the *binary model* is a good approximation of the quantized linear model. As will be

demonstrated later, efficient algorithms can be devised for the PU separation problem under the binary model.

## III. BINARY INFERENCE FRAMEWORK

In this section, we first discuss the identifiability of binary independent sources for the PUs from OR mixtures observed at the monitors, and then present an inference algorithm that determines the unknown mixing matrix and underlying sources.

### A. Identifiability

For an $m$-dimension binary random vector $\mathbf{x}$, the number of different realizations is $2^m$. From the data matrix $\boldsymbol{X}$, distribution of $\mathbf{x}$ can be estimated in a non-biased manner as the number of observations goes to infinity. We can initialize $n = 2^m - 1$ and $\boldsymbol{G}$ matrix of dimension $m \times (2^m - 1)$ with rows being all possible binary combinations of length $n$ (with exception of the all-0 entry). This results in a complete bipartite graph, in which an edge exists between any two vertices in $U$ (the set of monitors) and $V$ (the set of signal sources), respectively. For a random variable $y_j \in V$, its neighbors in $U$ is given by the non-zero entries in $g_j$ (i.e., the $j$th column of matrix $\boldsymbol{G}$). Thus, at most $2^m - 1$ independent components can be identified.

Define $p_l \triangleq \mathcal{P}(y_l = 1)$. Let the set

$$\boldsymbol{Y}(\mathbf{x}) = \left\{ \mathbf{y} \mid \bigvee_{j=1}^{n} (g_{ij} \wedge y_j) = x_i, \forall i \right\}.$$

Therefore,

$$
\begin{aligned}
\mathcal{P}(\mathbf{x}) &= \mathcal{P}(\mathbf{y} \in \boldsymbol{Y}(\mathbf{x})) = \sum_{\mathbf{y} \in \boldsymbol{Y}(\mathbf{x})} \mathcal{P}(\mathbf{y}) \\
&= \sum_{\mathbf{y} \in \boldsymbol{Y}(\mathbf{x})} \prod_{l=1}^{2^m-1} p_l^{y_l} (1 - p_l)^{1-y_l},
\end{aligned}
\tag{4}
$$

where $\mathcal{P}(\mathbf{y})$ is the joint probability of $\mathbf{y}$. The last equality holds due to the independence of $y_i$'s.

Given the distribution of random vectors $\mathbf{x} \in \{0,1\}^m$, $2^m - 1$ independent equations can be obtained from (4) due to the OR mixture model in (1). Since there are $2^m - 1$ unknowns (i.e., $p_l, l = 1, \ldots, 2^m - 1$), their values can be explicitly determined. Clearly, ambiguity exists if two or more independent sources have the same set of neighbors in $U$ (or equivalently, identical columns in $\boldsymbol{G}$). In this case, binary information is insufficient to distinguish these sources.

The set of equations in (4) are polynomials of sum product forms, which are difficult to solve. This necessitates the design of specialized algorithms. In the rest of the paper, abusing the notation a bit, we denote $\boldsymbol{G}$ the $m \times (2^m - 1)$ adjacency matrix for the bipartite graph $G' = (U, V', E')$, where $U = \{x_1, x_2, \ldots, x_m\}$ and $V' = \{y_1, y_2, \ldots, y_{2^m-1}\}$ (i.e. the set of all possible uniquely identifiable latent sources). Furthermore, we arrange $\boldsymbol{G}$ in the order such that $g_{kl} = 1$ if $(l \ll k) = 1$, for $k = 0, \ldots, m - 1$, where $\ll$ is the bit shift to the left. If the resulting $p_l = 0$ for some $l$, this implies the corresponding column $g_l$ can be removed from $\boldsymbol{G}$. For an example, consider

the network scenario with $m = 3$, the initialized $\boldsymbol{G}$ matrix will be:

$$\boldsymbol{G} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

with rows corresponding to 3 monitors and columns corresponding to 7 identifiable PUs.

### B. Inference algorithm

Before proceeding to the details of the proposed algorithm, we first present a few related technical lemmas as follows.

*Lemma 1:* Consider a set $\mathbf{x} = [x_1, x_2, \dots, x_{h-1}, x_h]^T$ generated by the data model in (1), i.e., $\exists$ binary independent sources $\mathbf{y}$, s.t., $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$. The conditional random vector $\mathbf{x}_{x_h=0} = [x_1, x_2, \dots, x_{h-1} | x_h = 0]^T$ corresponds to the vector of the first $h - 1$ elements of $\mathbf{x}$ when $x_h = 0$. Then, $\mathbf{x}_{x_h=0} = \boldsymbol{G}' \otimes \mathbf{y}'$, where $\boldsymbol{G}' = \boldsymbol{G}_{:,1,\dots,2^{h-1}}$ (i.e. the first $2^{h-1}$ columns of $\boldsymbol{G}$) and $\mathcal{P}(y_l' = 1) = \mathcal{P}(y_l = 1)$ for $l = 1, \dots, 2^{h-1}$.

*Proof:* We first derive the conditional probability distribution of the first $h - 1$ observation variables given $x_h = 0$,

$$
\begin{aligned}
& \mathcal{P}(x_1, x_2, \dots, x_{h-1} \mid x_h = 0) \\
= \ & \mathcal{P}(x_1, x_2, \dots, x_{h-1} \mid x_h = 0)\mathcal{P}(x_h = 0) \\
= \ & \sum_{\mathbf{y} \in \boldsymbol{Y}(\mathbf{x})} \prod_{l=1}^{2^h-1} p_l^{y_l}(1 - p_l)^{1-y_l} \\
= \ & \sum_{\substack{\mathbf{y}_{1..2^{h-1}} \in \boldsymbol{Y}(\mathbf{x}_{1..h-1}) \\ y_l = 0, \forall g_{hl} = 1}} \prod_{g_{hl}=0} p_l^{y_l}(1 - p_l)^{1-y_l} \prod_{g_{hl}=1}(1 - p_l)
\end{aligned}
$$

$$(5)$$

since $\mathcal{P}(x_h = 0) = \prod_{g_{hl}=1}(1 - p_l)$, we have

$$
\begin{aligned}
& \mathcal{P}(x_1, x_2, \dots, x_{h-1} \mid x_h = 0) \\
= \ & \sum_{\mathbf{y}' \in \boldsymbol{Y}(\mathbf{x}_{1:h-1})} \prod_{l=1}^{2^h-1} (p_l')^{y_l'}(1 - p_l')^{1-y_l'} \\
= \ & \sum_{\substack{\mathbf{y}_{1,\dots,2^{h-1}} \in \boldsymbol{Y}(\mathbf{x}_{1,\dots,h-1}) \\ y_l = 0, \forall g_{hl} = 1}} \prod_{g_{hl}=0} p_l^{y_l}(1 - p_l)^{1-y_l}.
\end{aligned}
$$

$$(6)$$

Clearly, by setting $\mathcal{P}(y_l' = 1) = \mathcal{P}(y_l = 1)$ for $l = 1, \dots, 2^{h-1}$, the above equality holds. In the other word, the conditional random vector $\mathbf{x}_{x_h=0} = \boldsymbol{G}' \otimes \mathbf{y}'$ for $\boldsymbol{G}' = \boldsymbol{G}_{:,1,\dots,2^{h-1}}$. ∎

The above lemma establishes that the conditional random vector $\mathbf{x}_{x_h=0}$ can be represented as an OR mixing of $2^{h-1}$ independent components. Furthermore, the set of the independent components is the same as the first $2^{h-1}$ independent components of $\mathbf{x}$ (under proper ordering).

Consider a sub-matrix of data matrix $\boldsymbol{X}$, $\boldsymbol{X}^0_{(h-1)\times T}$, where the rows correspond to observations of $x_1, x_2, \dots, x_{h-1}$ for $t = 1, 2, \dots, T$ such that $x_{ht} = 0$. Define $\boldsymbol{X}_{(h-1)\times T}$, which consists of the first $h - 1$ rows of $\boldsymbol{X}$. Suppose that we have computed the bICA for data matrices $\boldsymbol{X}^0_{(h-1)\times T}$ and

$\boldsymbol{X}_{(h-1)\times T}$. From Lemma 1, we know that $\boldsymbol{X}^0_{(h-1)\times T}$ is realization of OR mixtures of independent components, denoted by $\mathbf{y}^0_{h-1}$. Furthermore, $\mathcal{P}[\mathbf{y}^0_{2^{h-1}}(l) = 1] = \mathcal{P}(y_l = 1)$ for $l = 1, \dots, 2^{h-1}$. Clearly, $\boldsymbol{X}_{(h-1)\times T}$ is realization of OR mixtures of $2^{h-1}$ independent components, denoted by $\mathbf{y}_{2^{h-1}}$. Additionally, it is easy to see that the following holds:

$$
\begin{aligned}
& \mathcal{P}[\mathbf{y}_{2^{(h-1)}}(l) = 1] \\
= \ & 1 - [1 - \mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l) = 1)][1 - \mathcal{P}(y_{l+2^{h-1}} = 1)] \\
= \ & 1 - (1 - p_l)(1 - p_{l+2^{h-1}}),
\end{aligned}
$$

where $l = 1, \dots, 2^{h-1}$. Therefore,

$$
\begin{aligned}
p_l &= \mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l) = 1), & l = 1, \dots, 2^{h-1}, \\
p_{l+2^{h-1}} &= 1 - \frac{1 - \mathcal{P}(\mathbf{y}_{2^{h-1}}(l)=1)}{1 - \mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l)=1)}, & l = 2, \dots, 2^{h-1}, \\
p_{2^{h-1}+1} &= \frac{\mathcal{F}(x_h=1 \wedge x_i=0, \forall i \in [1\dots h-1])}{\prod_{l=1\dots 2^h, l \neq 2^{h-1}-1}(1 - p_l)}.
\end{aligned}
$$

$$(7)$$

The last equation above holds because realizations of $\mathbf{x}$ where ($x_k = 1$ while $x_i = 0; \forall i \in \{0, \dots, k-1\}$) are generated from OR mixtures of $\mathbf{y}_{2^{k-1}}$'s only. Define $\mathcal{F}(A)$ as the frequency of event $A$. To this end, we have the following iterative algorithm as illustrated in Algorithm 1.

When the number of monitors $m = 1$, there are only two possible unique sources, one that can be detected by the monitor, denoted by [1]; and one that cannot, denoted by [0]. Their active probabilities can easily be calculated by counting the frequency of $(x_1 = 1)$ and $(x_1 = 0)$ (lines 1 – 3). If $m \geq 2$, we apply Lemma 1 and (7) to estimate $p$ and $\boldsymbol{G}$ through a recursive process. We invoke FINDBICA on two sub-matrices $\boldsymbol{X}^0_{(m-1)\times T}$ and $\boldsymbol{X}_{(m-1)\times T}$ computed from $\boldsymbol{X}$ to determine $p_{1\dots2^{m-1}}$ and $p'_{1\dots2^{m-1}}$, then infer $p_{2^{m-1}+1\dots2^m}$ as in (7) (lines 6 – 8). Finally, $p_h$ and its corresponding column $g_h$ in $\boldsymbol{G}$ are pruned in the final result if $p_h < \varepsilon$ (lines 9 – 11).

---

**Algorithm 1:** Incremental binary ICA inference algorithm

FindBICA $(\boldsymbol{X})$
**input** : Data matrix $\boldsymbol{X}_{m\times T}$
**init** : $n = 2^m - 1$;
$p_h = 0, h = 1, \dots, n$;
$\boldsymbol{G} = m \times (2^m - 1)$ matrix with rows corresponding all possible binary vectors of length $m$;
$\varepsilon$ = the minimum threshold for $p_h$ to be considered a real component;

1 **if** $m = 1$ **then**
2     $p_1 = \mathcal{F}(x_1 = 0)$;
3     $p_2 = \mathcal{F}(x_1 = 1)$;
   **else**
4     $p_{1\dots2^{m-1}}$ = FindBICA $(\boldsymbol{X}^0_{(m-1)\times T})$;
5     $p'_{1\dots2^{m-1}}$ = FindBICA $(\boldsymbol{X}_{(m-1)\times T})$;
6     **for** $l = 2, \dots, 2^{m-1}$ **do**
7       $p_{l+2^{m-1}} = 1 - \frac{1-p_l'}{1-p_l}$;
8     $p_{2^{m-1}+1} = \frac{\mathcal{F}(x_m=1 \wedge x_i=0, \forall i \in [1\dots m-1])}{\prod_{l=1\dots2^m-1, l \neq 2^{m-1}+1}(1 - p_l)}$;

9 **for** $h = 1, \dots, 2^m$ **do**
10     **if** $(p_h < \varepsilon) \vee (p_h = 0)$ **then**
11       prune $p_h$ and corresponding columns $g_h$;

12 **output**: $p$ and $\boldsymbol{G}$

---

**Computation complexity:** Let $S(m)$ be the computation time for finding bICA given $\boldsymbol{X}_{m\times T}$. From Algorithm 1, we have,

$$S(m) = 2S(m-1) + c2^m, \tag{8}$$

where $c$ is a constant. It is easy to verify $S(m) = cm2^m$. Therefore, Algorithm 1 has an exponential computation complexity with respect to $m$. This is clearly undesirable for large $m$'s. However, we notice that in practice, correlations among $x_i$'s exhibit locality, and matrix $\boldsymbol{G}$ tends to be sparse. Instead of using a complete bipartite graph to represent $\boldsymbol{G}$, the degree of vertices in $V$ (or the number of non-zero elements in $g_k$) tends to be much less than $m$. More specifically, for every pair of monitors $i$ and $k$, we compute the covariance between their observations:

$$cov(i,k) = \frac{\sum_t^T x_{it}x_{kt}}{T} - \frac{\sum_t^T x_{it}}{T}\frac{\sum_t^T x_{kt}}{T}. \tag{9}$$

If $cov(i,k) < \epsilon$, where $\epsilon$ is a small value (e.g., the upper confidence bound of $cov(i,k)$ estimate), we can remove the corresponding columns in $\boldsymbol{G}$ and elements in $\mathbf{y}$.

## IV. THE INVERSE PROBLEM

Now we have the mixing matrix $\boldsymbol{G}_{m\times n}$ and the active probabilities $\mathcal{P}(\mathbf{y})$, given observation $\boldsymbol{X}_{m\times T}$, the inverse problem concerns inferring the realizations of the latent variables $\boldsymbol{Y}_{n\times T}$. Extracting multiple PUs' activities from the OR mixture observations is a challenging but important problem in cognitive radio networks. Interesting information, such as the PU channel usage pattern can be inferred once $\boldsymbol{Y}$ is available. The SUs will then be able to adopt better spectrum sensing and access strategies to exploit the spectrum holes more effectively.

Recall that $n$ is the number of PUs (latent variables). Denote $y_i$ a binary variable for the $i$th latent variable. Let $\mathbf{x} = \boldsymbol{G}\otimes\mathbf{y}$. We assume that the probability of observing $\boldsymbol{X}$ given $\mathbf{x}$ depends on their Hamming distance $d(\mathbf{x},\boldsymbol{X}) = \sum_i |\boldsymbol{X}_i - x_i|$, and $\mathcal{P}(\mathbf{x}|\boldsymbol{X}) = p_e^{d(\mathbf{x},\boldsymbol{X})}(1-p_e)^{m-d(\mathbf{x},\boldsymbol{X})}$, where $p_e$ is the error probability of the binary symmetric channel. To determine $\mathbf{y}$, we can maximize the posterior probability of $\mathbf{y}$ given $\boldsymbol{X}$ derived as follows,

$$\begin{aligned}\mathcal{P}\{\mathbf{y}|\boldsymbol{X}\} &= \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}}\\ &= \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}}\\ &\overset{(a)}{=} \frac{\mathcal{P}\{\boldsymbol{X},\mathbf{x}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}}\\ &\overset{(b)}{=} \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{x}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}}\\ &= \frac{\prod_{i=1}^m \mathcal{P}\{\boldsymbol{X}_i|x_i\}\prod_{j=1}^n \mathcal{P}\{y_i\}}{\mathcal{P}\{\boldsymbol{X}\}}\\ &= \frac{\prod_{i=1}^m p_e^{|x_i-\boldsymbol{X}_i|}(1-p_e)^{1-|x_i-\boldsymbol{X}_i|}\prod_{j=1}^n p_i^{y_i}(1-p_i)^{1-y_i}}{\mathcal{P}\{\boldsymbol{X}\}}\end{aligned}$$

where $\mathbf{x} = \boldsymbol{G}\otimes\mathbf{y}$. $(a)$ and $(b)$ are due to the deterministic relationship between $\mathbf{x}$ and $\mathbf{y}$. Recall that $x_i = \bigvee_{j=1}^n (g_{ij}\wedge y_j), i = 1,\ldots,m$. With $M$ is a "large enough" constant, we can use big-$M$ formulation [12] to relax the disjunctive set and convert the above relationship between $\mathbf{x}$ and $\mathbf{y}$ into the following two sets of conditions:

$$\begin{aligned}x_i &\le \sum_{j=1}^n g_{ij}y_j, \quad \forall i=1,\ldots,m.\\ M\cdot x_i &\ge \sum_{j=1}^n g_{ij}y_j, \quad \forall i=1,\ldots,m.\end{aligned} \tag{10}$$

Here, since $\sum_{j=1}^n g_{ij}y_j \le n$, we can set $M = n$. Finally, taking $\log$ on both sides and introducing additional auxiliary variable $\alpha_i = |\boldsymbol{X}_i - x_i|$, we have the the following integer programming problem:

$$\begin{aligned}\max_{\alpha,y}. \quad &\sum_{i=1}^m [\alpha_i\log p_e + (1-\alpha_i)\log(1-p_e)]\\ &+\sum_{j=1}^n [(1-y_j)\log(1-p_j) + y_j\log p_j]\\ \text{s.t.} \quad &x_i \le \sum_{j=1}^n g_{ij}y_j, \quad \forall i=1,\ldots,m,\\ &n\cdot x_i \ge \sum_{j=1}^n g_{ij}y_j, \quad \forall i=1,\ldots,m,\\ &\alpha_i \ge \boldsymbol{X}_i - x_i, \quad \forall i=1,\ldots,m,\\ &\alpha_i \ge x_i - \boldsymbol{X}_i, \quad \forall i=1,\ldots,m,\\ &\alpha_i, x_i, y_j = \{0,1\}, \quad \forall i=1,\ldots,m, j=1,\ldots,n.\end{aligned} \tag{11}$$

This optimization function can be solved using ILP solvers. Note that $p_e$ can be thought of the penalty for mismatches between $x_i$ and $\boldsymbol{X}_i$.

**Zero Error Case:** If $\boldsymbol{X}$ is perfectly observed, containing no noise, we have $p_e = 0$ and $\alpha_i = \mathbf{x}_i - \boldsymbol{X}_i = 0$, or equivalently, $\mathbf{x}_i = \boldsymbol{X}_i$. The integer programming problem in (11) can now be simplified as:

$$\begin{aligned}\max_y. \quad &\sum_{j=1}^n [(1-y_j)\log(1-p_j) + y_j\log p_j]\\ \text{s.t.} \quad &\boldsymbol{X}_i \le \sum_{j=1}^n g_{ij}y_j, \forall i=1,\ldots,m,\\ &n\cdot\boldsymbol{X}_i \ge \sum_{j=1}^n g_{ij}y_j, \forall i=1,\ldots,m,\\ &y_j = \{0,1\}, \forall j=1,\ldots,n.\end{aligned} \tag{12}$$

Clearly, the computation complexity of the zero error case is lower compared to (11). It can also be used in the case where prior knowledge regarding the noise level is not available.

## V. EVALUATION

In this section, we first introduce the performance metrics, and then present evaluation results on a synthetic data set varying the number of PUs.

### A. Performance metrics

We denote by $\hat{p}$ and $\hat{\boldsymbol{G}}$ the inferred active probability of PUs and the inferred mixing matrix, respectively.

*1) Structure Error Ratio:* This metric indicates how accurate the mixing matrix is estimated. It is defined by the Hamming distance between $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ divided by its size.

$$\bar{H}_g \overset{\triangle}{=} \frac{1}{mn}\sum_{i=1}^n d^H(g_i,\hat{g}_i). \tag{13}$$

To estimate $\bar{H}_g$ however, two challenges remain: First, the number of inferred independent components may not be identical as the ground truth. Second, the order of independent components in $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ may be different.

---

**Algorithm 2:** Bipartite graph matching algorithm

---

MatchPG $(\boldsymbol{G}, \hat{\boldsymbol{G}}, \hat{p})$
  **input** : $\boldsymbol{G}_{m \times n}, \hat{\boldsymbol{G}}_{m \times \hat{n}}, \hat{p}_{1 \times \hat{n}}; (n \le \hat{n} \le 2^m)$
  **init** : $\hat{\boldsymbol{G}}'_{m \times \hat{n}} = 0; \hat{p}'_{1 \times \hat{n}} = 0; \boldsymbol{C}_{\hat{n} \times \hat{n}} = 0;$

**1**  **for** $i = 1, \dots, \hat{n}$ **do**
**2**    **for** $j = 1, \dots, \hat{n}$ **do**
**3**      **if** $g_i = 0$ **then**
**4**        $c_{ij} = d^H(g_i, \hat{g}_j) \times m;$
      **else**
**5**        $c_{ij} = d^H(g_i, \hat{g}_j);$

**6**  $\boldsymbol{A}$ = BipartiteMatching $(\boldsymbol{C})$;
**7**  **for** $i = 1, \dots, \hat{n}$ **do**
**8**    find $j$ such that $a_{ij} = 1$;
**9**    $\hat{g}'_i = \hat{g}_j$;
**10**   $\hat{p}'_i = \hat{p}_j$;
**11** Prune $\hat{\boldsymbol{G}}'$: $\hat{\boldsymbol{G}}' = \hat{g}'_{1 \dots n}$;
**12** Prune $\hat{p}'$: $\hat{p}' = \hat{p}'_{1 \dots n}$;
**13** **output**: $\hat{\boldsymbol{G}}'$ and $\hat{p}'$

---



Fig. 3: From top to bottom: inferred matrix $\hat{\boldsymbol{G}}$ with 18 inferred components, transformed matrix $\hat{\boldsymbol{G}}'$ with only 10 components remaining (8 noisy ones were removed), original $\boldsymbol{G}$ and difference matrix between $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}'$. Black dot = 1 and white dot = 0.

To solve the first problem, we can either prune $\hat{\boldsymbol{G}}$ or introduce columns into $\boldsymbol{G}$ to equalize the number of components ($n = \hat{n}$, where $\hat{n}$ is the number of columns in $\hat{\boldsymbol{G}}$). For the second problem, we propose a matching algorithm that minimizes the Hamming distance between $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ by permuting the corresponding columns in $\hat{\boldsymbol{G}}$.

**Structure Matching Problem:** A naive matching algorithm needs to consider all $\hat{n}!$ column permutations of $\hat{\boldsymbol{G}}$, and chooses the one that has the minimal Hamming distance to $\boldsymbol{G}$. This approach incurs an exponential computation complexity. Next, we first formulate the best match as an ILP problem. Denote the Hamming distance between column $\hat{g}_i$ and $g_j$ as $c_{ij} \ge 0$. Define a permutation matrix $\boldsymbol{A}_{n \times n}$ with $a_{ij} = 1$ indicating that the $i$th column in $\hat{\boldsymbol{G}}$ is matched with the $j$th column in $\boldsymbol{G}$. The problem now is to find a permutation matrix such that the total Hamming distance between $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ (denoted by $d^H(\boldsymbol{G}, \hat{\boldsymbol{G}})$) is minimized. We can formulate this problem as an ILP as follows:

$$
\begin{aligned}
\min_a . \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} a_{ij} \\
\text{s.t.} \quad & \sum_{i=1}^{n} a_{ij} = 1, \\
& \sum_{j=1}^{n} a_{ij} = 1, \\
& a_{ij} = 0, 1 \; \forall i, j = 1, \dots, n.
\end{aligned}
\tag{14}
$$

The constraints ensure the resulting $\boldsymbol{A}$ is a permutation matrix. This problem can be solved using ILP solvers. However, we observe that the ILP is equivalent to a maximum-weight bipartite matching problem. In the bipartite graph, the vertices are positions of the columns, and the edge weights are the Hamming distance of the respective columns. If we consider $d^H(g_i, \hat{g}_i)$, the Hamming distance between column $g_i$ and $\hat{g}_i$, to be the "cost" of matching $\hat{g}_i$ to $g_i$, then the maximum-weight bipartite matching problem can be solved in $O(n^3)$
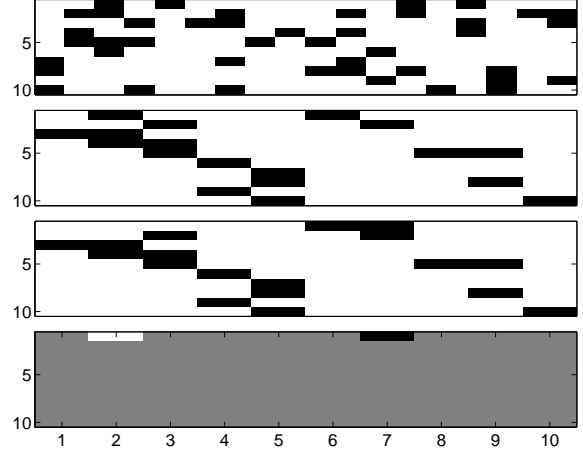
running time [13], where $n$ is the number of vertices. The algorithm requires $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ to have the same number of columns.

One greedy solution is to prune $\hat{\boldsymbol{G}}$ by selecting the top $n$ components from $\hat{\boldsymbol{G}}$, which have the highest associated probabilities $\hat{p}_i$ since they are the most likely true components. However, when $T$ is small and/or under large noise, we may not have sufficient observations to correctly identify components in $\boldsymbol{G}$ with high confidence. As a result, true components might have lower active probabilities comparing to the noise components. To address the problem, we instead keep a larger $\hat{n}$ and introduce $\hat{n} - n$ artificial components into $\boldsymbol{G}$. These components will be represented by zero columns in $\boldsymbol{G}$. While matching the inferred columns in $\hat{\boldsymbol{G}}$ to the columns in $\boldsymbol{G}$, clearly an undesirable scenario occurs when we accidently match a column in $\hat{\boldsymbol{G}}$ to an additive zero column in $\boldsymbol{G}$. This happens when an inferred column $\hat{g}_i$ is sparse (i.e. having a very small Hamming distance to the zero column). To avoid the incident, we multiply the cost of matching any column in $\hat{\boldsymbol{G}}$ to a zero column in $\boldsymbol{G}$ by $m$. This eliminates the case in which a column $\hat{g}_i$ is matched with a zero column in $\boldsymbol{G}$, since it is more expensive than matching with another non-zero column $g_i$. We can now select the best $n$ candidates in $\hat{\boldsymbol{G}}$, which yields a reduced mixing matrix $\hat{\boldsymbol{G}}'$ of size $m \times n$, and elements in active probability vector $\hat{p}'$ will also be selected accordingly. The solution to the structure matching problem is detailed in Algorithm 2.

In the algorithm, lines 1 – 5 build the input weight matrix $\boldsymbol{C}_{\hat{n} \times \hat{n}}$ for the bipartite matching algorithm. If $g_i$ is a zero column, $c_{ij}$ will be scaled by $m$ to avoid the matching between column $g_i$ and $\hat{g}_j$ (line 4). The bipartite matching algorithm finds the optimal permutation matrix $\boldsymbol{A}$ to transform $\hat{\boldsymbol{G}}$ into $\hat{\boldsymbol{G}}'$ that is "closest" to $\boldsymbol{G}$ (lines 6 – 10). We are only interested in the first $n$ columns of $\hat{\boldsymbol{G}}'$ and $\hat{p}'$ (as they most likely represent the true PUs). Therefore, $\hat{\boldsymbol{G}}'$ and $\hat{p}'$ are pruned in lines 11 – 13.

As an example, the inferred result of a random network with $n = m = 10$ is given in Figure 3. Non-zero entries and zero entries of $G$, $\hat{G}$, and $\hat{G}'$ are shown as black and white dots, respectively. The entry-wise difference matrix $|G - \hat{G}'|$ is given in the bottom graph. Gray dots in the difference matrix indicate identical entries in the inferred $\hat{G}$ and the original $G$; and black dots indicate different entries (and thus errors in the inferred matrix). In this case, only the first row (corresponding to the first monitor $x_1$) contains some errors.

*2) PU Active Probability Error Ratio:* Prediction error in the inferred active probabilities of PUs is measured by the root mean square error ratio between $\hat{p}'$ and $p$, defined as follows:

$$\bar{P} \triangleq \sqrt{\frac{\sum_{i=1}^{n}(\hat{p}_i' - p_i)^2}{n} / \frac{\sum_{i=1}^{n} p_i}{n}}. \tag{15}$$

PU active probability error ratio can be interpreted as the fraction of the inferred active probability that deviates from the true values.

*3) Miscount in the number of PUs:* Accuracy of the inference algorithm can also be assessed by evaluating the difference between the number of inferred PUs $\hat{n}$ and the real number of PUs $n$ in the system. Clearly, with a smaller threshold value $\varepsilon$ in Algorithm 1, the number of inferred PUs $\hat{n}$ may increase. In the subsequent experiments, we fix $\varepsilon = 0.01$ and evaluate the changes in $\hat{n}$ as the real number of PUs increases from 5 to 20.

*4) PU Activity Error Ratio:* After applying FINDBICA in Algorithm 1 on the measurement data of length $T$ to obtain $\hat{G}$ and $\hat{p}$, realizations of the hidden causes (i.e. PUs) can be computed by solving the maximum likelihood estimation problem in (11). We define

$$\bar{H}_y \triangleq \frac{1}{nT} \sum_{i=1}^{T} d^H(y_i, \hat{y}_i), \tag{16}$$

where $y_i$ is the $i$'th column of $Y$. Similar to $\bar{H}_g$, this metric measures how accurately the PUs' activity matrix is inferred by calculating the ratio between the size of $\mathbf{y}$ and the absolute difference between $\mathbf{y}$ and $\hat{\mathbf{y}}$.

### B. Experimental results

For evaluation, we consider the same simulation scenario as in Section II-A, where 10 monitors are randomly deployed to monitor $n$ PUs using a single channel. Algorithms are implemented in Matlab, and all experiments are conducted on a workstation with an Intel Core 2 Duo T5750@2.00GHz processor and 2GB RAM. Noise is introduced by randomly flip a bit in the observation matrix $X$ from 1 to 0 (and vice versa) at probability $e$. $e$ is set at 0%, 2%, and 5% in our simulations. All presented results are averages of 50 runs with different initializations. In the experiments, we vary the number of PUs $n$ and the number of observations $T$.

*1) Varying the number of PUs:* In the first set of experiments, we fix the sample size $T = 10,000$ and vary the number of PUs from 5 to 20 to study its impact on the accuracy of our method. Experiment results over 50 runs for each PU setting are shown in Figure 4. In absence of noise, we observe that the inferred mixing matrix $\hat{G}$ is mostly correct even for a large number of PUs (Figure 4(a)). As the number of

PUs increases, errors in the inferred active probabilities and the inferred number of PUs tend to increase though within 10% and 0.65 as shown in Figure 4(b) and (c). Recall that the PUs are ordered based on their respective columns in $\hat{G}$. Therefore, errors in $\hat{G}$ may have a cascading effect on PU active probability error ratio since we may compare the wrong pair of PUs as a result. Performance of the algorithm tends to degrade with more noises. In particular, as shown in Figure 4(c), more components are inferred compared to the ground truth. This is because when the noise probability $p_e$ is greater than the threshold value, some noise components are erroneously introduced.

The errors in determining the set of active PUs are shown in Figure 4(d). We can see again that the proposed algorithm achieves remarkable accuracy at zero-noise level. Prediction error on $Y$ is only 1% for 5 PUs, and gradually increases up to 5% for 20 PUs. At 2% and 5% noise levels, performance degrades as the prediction error goes up to about 10%. Noise has two effects on the solution to the inverse problem. First, the inferred mixing matrix and active probability can be erroneous. Second, no maximum likelihood estimator guarantees to give the exact result when the problem is under or close to being under-determined with noisy measurements. In fact, we have experimented with the case in which $G$ and $p$ are both known, the results are similar. This implies that the main source of errors in solving the inverse problem comes from the problem itself being under-determined or close to being under-determined.

Finally, as shown in Figure 4(e), the computation time of proposed algorithms is negligible, mostly under 0.2 second without noise. With noise, computation time increases but is similar at the two noise levels (differing by less than 0.5 second). The presence of noise may introduce noise components and render the estimation of correlation (in Equation (9)) inaccurate. Thus, higher processing time entails.

*2) Varying the size of observations $T$:* In the second set of experiments, we fix the number of PUs $n = 10$ and study the impact of the observation size $T$. A small $T$ (and thus insufficient observations) would lead to higher uncertainty while a large $T$ incurs higher computation overhead. Furthermore, if $T$ is too small, some PUs may never be active in the trace, making them impossible to be inferred. It is therefore interesting to investigate the effect of $T$ on the accuracy and computation overhead of the proposed algorithm.

Experiment results over 50 runs for each observation size are shown in Figure 5. As expected, the structure error ratio and the PU active probability error ratio reduce significantly as $T$ increases from 50 to 1,000 (Figure 5(a) and (b)). If we further increase $T$ from 1,000 to 10,000, the performance gain is somewhat marginal. However, the computation time grows considerably since it takes longer to process the observations (Figure 5(e)). From Figure 5(c) and (d), we also see that the miscount of PUs and the PU activity error ratio are not sensitive to the sample size $T$, but are more affected by the noise level.

(a) Structure error ratio

(b) PU active probability error ratio

(b) Miscount of PUs

(d) PU activity error ratio
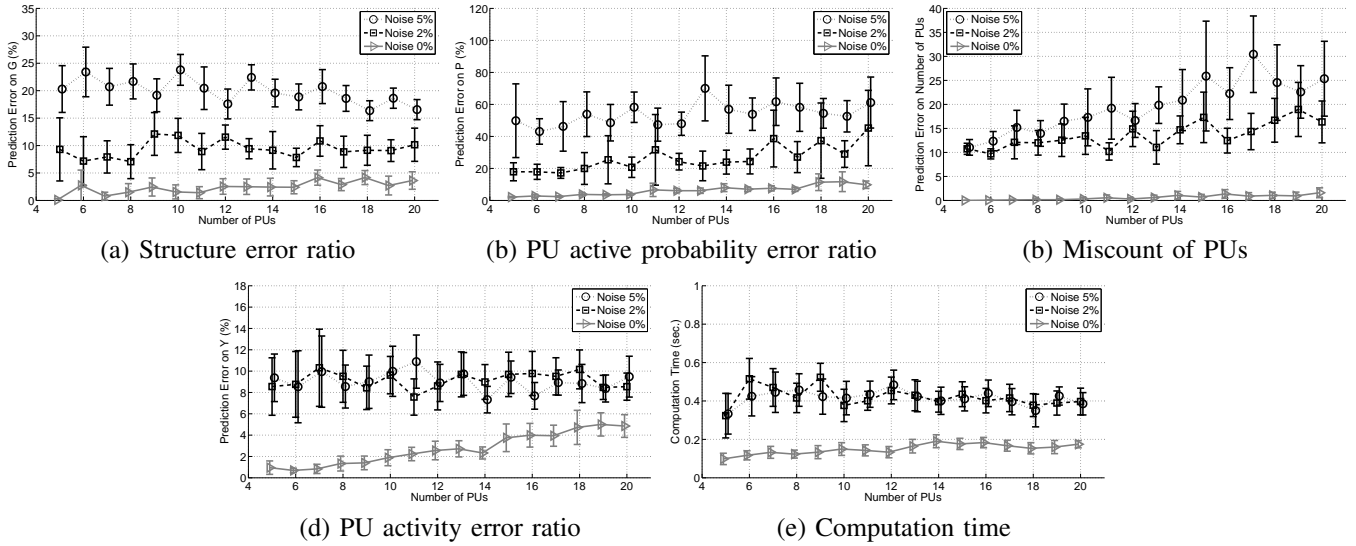
(e) Computation time

Fig. 4: Effects of noise level and the number of PUs on inference results. Noise is introduced at three levels: 0%, 2% and 5%. Results are averages of 50 runs with different initial seed. Symmetric error bars indicate standard deviations.
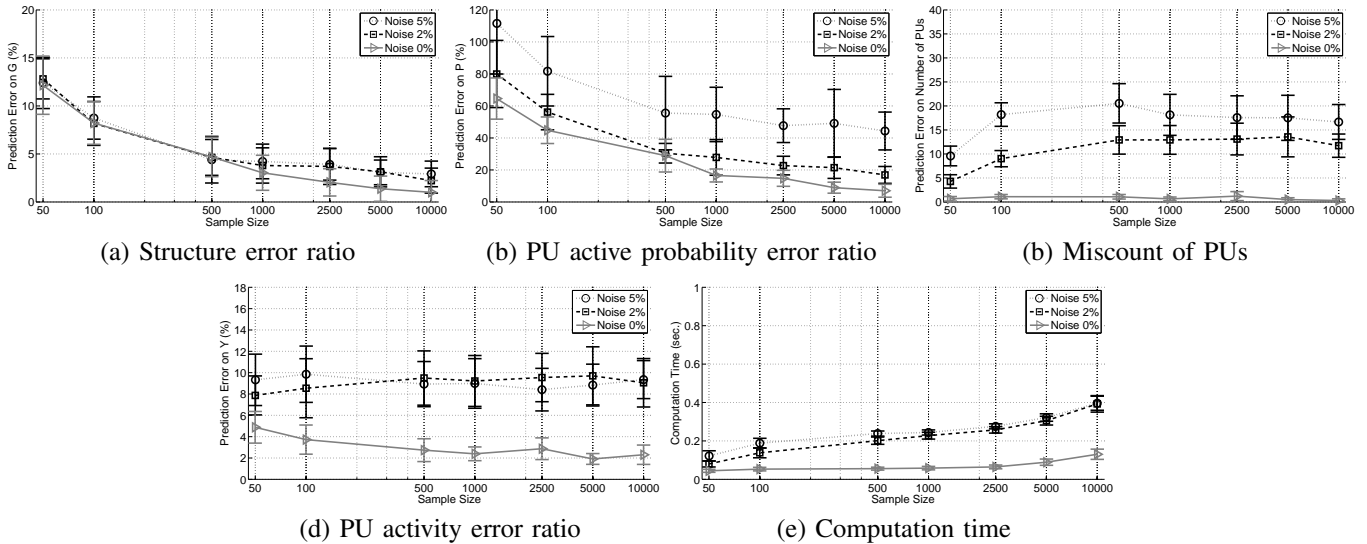


(a) Structure error ratio

(b) PU active probability error ratio

(b) Miscount of PUs

(d) PU activity error ratio

(e) Computation time

Fig. 5: Effects of noise level and the size of observations on inference results. The $x$-axis is in logarithmic scale.

## VI. RELATED WORK

Independent component analysis (ICA) has been studied in the past as a computational method for separating a multivariate signal into additive subcomponents supposing the mutual statistical independence of the non-Gaussian source signals. Most ICA methods assume linear mixing of continuous signals [11]. A special variant of ICA, called Boolean Independent Component Analysis (BICA), considers boolean mixing (e.g., OR, XOR etc.) of binary signals. Existing solutions to BICA mainly differ in their assumptions of prior distribution of the mixing matrix, noise model, and/or hidden causes. In [14], Yeredor considers BICA in XOR mixtures and investigates the identifiability problem. A deflation algorithm is proposed for source separation based on entropy minimization. In [14] the number of independent random sources $K$ is assumed to be known. Furthermore, the mixing matrix is an $K$-by-

$K$ invertible matrix. In [15], infinite number of hidden causes following the same Bernoulli distribution are assumed. Reversible jump Markov chain Monte Carlo and Gibbs sampler techniques are applied. In contrast, in our model, the hidden causes may follow different distribution and the mixing matrix tends to be sparse. Streich *et al.* [16] study the problem of multi-assignment clustering for boolean data, where the observations either from a signal following OR mixtures or from a noise component. The key assumption made in this work is that the elements of matrix $\boldsymbol{X}$ are conditionally independent given the model parameters. This greatly reduces the computational complexity and makes the scheme amenable to gradient descent optimization solution. This assumption is in general invalid. In [17], the problem of factorization and denoise of binary data due to independent continuous sources is considered, which follow beta distribution. Finally, [15]

consider under-presented case of less sensors than sources with continuous noise, while [17] and [16] deal with over-determined case, where the number of sensors is much larger. In this work, we consider primarily the under-presented cases encountered in data networks.

There exists a large body of work on blind deconvolution with binary sources in the context of wireless communication [18], [19]. In time-invariant linear channels, the output signal $x(k)$ is a convolution of the channel realizations $a(k)$ and the input signal $s(k)$, $k = 1, 2, \ldots, K$ as follows:

$$x(k) = \sum_{l=0}^{L} a(l)s(k-l), k = 1, \ldots, K. \qquad (17)$$

The objective is to recover the input signal $s$. Both stochastic and deterministic approaches have been devised for blind deconvolution. As evident from (17), the output signals are linear mixtures of the input sources in time, and additionally the mixture model follows a specific structure.

Literature on boolean/binary factor analysis (BFA) is also related to our work. The goal of BFA is to decompose a binary matrix $\boldsymbol{X}_{m \times T}$ into $\boldsymbol{A}_{m \times n} \otimes \boldsymbol{B}_{n \times T}$ with $\otimes$ being the *OR* mixture relationship as defined in (1). We use the same notation of $m$, $n$, and $T$ to illustrate the relationship between BFA and bICA. $\boldsymbol{X}$ in BFA is often called an attribute-object matrix providing $m$-dimension attributes of $T$ objects. $\boldsymbol{A}$ and $\boldsymbol{B}$ are the attribute-factor and factor-object matrices. All the elements in $\boldsymbol{X}$, $\boldsymbol{A}$, and $\boldsymbol{B}$ are either 0 or 1. $n$ is defined to be the number of underlying factors and is assumed to be considerably smaller than the number of objects $T$. BFA methods aim to find a feasible decomposition minimizing $n$. Frolov *et al.* study the problem of factoring a binary matrix in a series of papers [20], [21], [22] using Hopfield neural networks. This approaches are based on heuristics and do not provide much theoretical insight regarding the properties of the resulting decomposition. More recently, Belohlavek *et al.* propose a matrix decomposition method utilizing formal concept analysis [23]. The paper claims that optimal decomposition with the minimum number of factors are those where factors are formal concepts. It is important to note that even though BFA assumes a similar disjunctive mixture model to our problem, the objective is different. While BFA tries to find a matrix factorization so that the number of factors are minimized, bICA tries to identify independent components. One can easily come up an example, where the number of independent components (factors) is larger than the number of attributes, while BFA always finds factors no larger than the number of attributes.

## VII. CONCLUSIONS

In this paper, we introduced the PU separation problem for cognitive radio networks and argue its relevance in collaborative spectrum sensing and monitor resource allocation. We demonstrated that a binary mixing model is sufficient to characterize the behavior of energy detectors in presence of multiple PUs, and devised a binary inference framework to resolve the PU separation problem. The results are somewhat surprising that PUs can be accurately separated and identified with only binary observations from the set of monitors to which they are observable. Simulation validation shows that the PU-SU relationship as well as the PUs' statistics and activities can be estimated with high accuracy when the noise is marginal.

## REFERENCES

[1] J. Mitola and G. Q. Maguire, "Cognitive radio: Making software radios more personal," *IEEE Pers. Commun.*, vol. 6, pp. 13–18, Aug. 1999.

[2] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 2, pp. 201 – 220, 2005.

[3] D. Niyato, E. Hossein, and Z. Han, *Dynamic spectrum access in cognitive radio networks*. Cambridge, UK: Cambridge University Press, 2009.

[4] A. Ghasemi and E. S. Sousa, "Collaborative spectrum sensing for opportunistic access in fading environments," in *Proc. of IEEE Symp. New Frontiers in Dynamic Spectrum Access Networks*, Baltimore, USA, Nov 2005.

[5] C. Sun, W. Zhang, and K. B. Letaief, "Cooperative spectrum sensing for cognitive radios under bandwidth constraint," in *Proc. of IEEE Wireless Communications and Networking Conf.*, Hong Kong, China, Mar 2007.

[6] P. Houz, S. B. Jemaa, and P. Cordier, "Common pilot channel for network selection," in *Proc. of IEEE Vehicular Technology - Spring*, Melbourne, Australia, May 2006.

[7] M. Filo, A. Hossain, A. R. Biswas, and R. Piesiewicz, "Cognitive pilot channel: Enabler for radio systems coexistence," in *Proc. of Second Int. Workshop on Cognitive Radio and Advanced Spectrum Management*, alborg, Denmark, May 2009.

[8] O. Sallent, J. Perez-Romero, R. Agusti, and P. Cordier, "Cognitive pilot channel enabling spectrum awareness," in *Proc. Int. Conf. on Communications, Workshop on*, Dresden, Germany, Jun. 2009.

[9] J. Perez-Romero, O. Salient, R. Agusti, and L. Giupponi, "A novel on-demand cognitive pilot channel enabling dynamic spectrum allocation," in *Proc. of IEEE Int. Symp. on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, Dublin, Ireland, Apr. 2007.

[10] W. Lina, "Novel scheme for cooperation spectrum sensing in cognitive radio networks," in *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on*, vol. 4, 2010, pp. 371 –375.

[11] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Netw.*, vol. 13, no. 4-5, pp. 411–430, 2000.

[12] I. Griva, S. G. Nash, and A. Sofer, *Linear and Nonlinear Optimization, Second Edition*, 2nd ed. Society for Industrial Mathematics, December 2008.

[13] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[14] A. Yeredor, "Ica in boolean xor mixtures," in *Proceedings of the 7th international conference on Independent component analysis and signal separation*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 827–835.

[15] F. W. Computer and F. Wood, "A non-parametric bayesian method for inferring hidden causes," in *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2006, pp. 536–543.

[16] A. P. Streich, M. Frank, D. Basin, and J. M. Buhmann, "Multi-assignment clustering for boolean data," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 969–976.

[17] A. Kabn and E. Bingham, "Factorisation and denoising of 0-1 data: A variational approach," *Neurocomputing*, vol. 71, no. 10-12, pp. 2291 – 2308, 2008, neurocomputing for Vision Research; Advances in Blind Signal Processing.

[18] K. Diamantaras and T. Papadimitriou, "Blind deconvolution of multi-input single-output systems with binary sources," *Signal Processing, IEEE Transactions on*, vol. 54, no. 10, pp. 3720 –3731, October 2006.

[19] Y. Li, A. Cichocki, and L. Zhang, "Blind separation and extraction of binary sources," *IEICE Trans. Fund. Electron. Commun. Comput. Sci.*, vol. E86-A, no. 3, p. 580589, 2003.

[20] D. Húsek, H. Rezanková, V. Snásel, A. A. Frolov, and P. Polyakov, "Neural network nonlinear factor analysis of high dimensional binary signals," in *SITIS*, 2005, pp. 86–89.

[21] A. A. Frolov, D. Húsek, I. P. Muraviev, and P. Y. Polyakov, "Boolean factor analysis by attractor neural network," *IEEE Transactions on Neural Networks*, vol. 18, no. 3, pp. 698–707, 2007.

[22] D. Húsek, P. Moravec, V. Snásel, A. A. Frolov, H. Rezanková, and P. Polyakov, "Comparison of neural network boolean factor analysis method with some other dimension reduction methods on bars problem," in *PReMI*, 2007, pp. 235–243.

[23] R. Belohlavek and V. Vychodil, "Discovery of optimal factors in binary data via a novel method of matrix decomposition," *J. Comput. Syst. Sci.*, vol. 76, pp. 3–20, February 2010.