Creating Spatio-temporal Spectrum Maps from Sparse Crowdsensed Data

Md. Shaifur Rahman, Himanshu Gupta, Ayon Chakraborty[†] and Samir Das
Stony Brook University, NY. [†]NEC Lab, USA.

***Abstract*—Shared spectrum systems is an emerging paradigm to improve spectrum utilization and thus address the unabated increase in mobile data consumption. The paradigm allows the "unused" spectrum bands of licensed Primary Users (PUs) to be shared with Secondary Users (SUs), without causing any harmful interference to the PUs. Allocation of spectrum to the SUs is done based on spectrum availability at the SUs' locations; such allocation of spectrum is greatly facilitated by *spectrum occupancy maps*. In this work, we address the problem of creating spectrum occupancy maps from spectrum occupancy data over a large number of instants, in the challenging scenario of dynamically (temporally) changing spectrum occupancy due to intermittent transmission of primary users. The problem is particularly challenging when the available occupancy data is very sparse spatially, i.e., only very few locations report sensing data at any particular instant. We design various techniques to create spectrum maps in the above context, including a promising correlation-based merging method that merges observation vectors iteratively in conjunction with careful interpolation. Using extensive simulation over data including real data from cellular and deployed WiFi settings, we show that the correlation-based method is very effective in generating high-accuracy spatio-temporal spectrum maps even with very sparse observation vectors (as long as the number of such vectors is large enough).**

I. Introduction

Radio frequency (RF) spectrum is a natural resource in great demand, due to our unabated increase in mobile (and hence, wireless) data consumption. Some projections estimate a 1000-fold increase in capacity within a decade [1]. Researchers have been addressing this impending capacity crunch using various mechanisms — one of most effective being development of shared spectrum paradigm (e.g., white spaces). The paradigm improves the spectrum utilization efficiency by allowing unlicensed/secondary users (SUSE) to exploit unused spectrum bands of licensed primary users (PUs). That is, SUs can opportunistically use unused spectrum bands (aka spectrum holes or white spaces) as long as their use does not cause harmful (wireless) interference to PUs. This is typically facilitated by a (centralized) spectrum manager (SM) who allocates available spectrum to a request SU based on available information about spatio-temporal spectrum occupancy or availability information.

To facilitate the above shared spectrum paradigms, there is a need to deduce or generate spectrum occupancy maps that can be used to allocate available (i.e., without causing interference to the PUs). Fundamentally, there are two ways to do this: (i) Use PUs' information such as location, transmit power, intended receivers, etc. along with assumption of a wireless

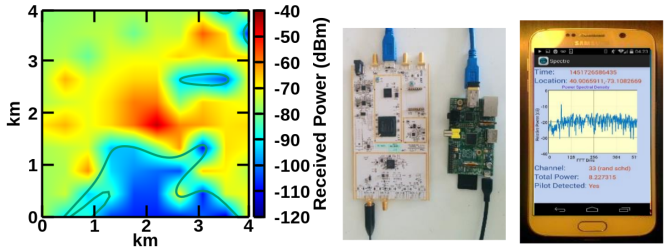

Fig. 1. (a) Spectrum map generated from sensor data (b) Commodity spectrum sensors used for our data-collection

propagation (signal attenuation) model [2] to deduce spectrum power that can be allocated at a request location, (ii) Use sensing reports from geographically distributed spectrum sensors (SS nodes) to deduce spectrum occupancy map, and allocated spectrum at locations with spectrum availability in the map. The first strategy suffers from obvious shortcomings—need to assume a wireless propagation model, and in many applications, PU information is not available, e.g., for strategic PUs such as naval carriers in 3.5GHz band in US where the privacy of these PUs is fully preserved [3]. In this paper, we consider the second model, wherein spectrum power to be allocation is estimated based on the spectrum occupancy map generated from sensing reports of geographically distributed spectrum sensors (SS). In particular, to facilitate high granularity spectrum data collection via relatively inexpensive means, we consider crowdsourced spectrum sensors. The practicality of crowdsourcing has been demonstrated in research projects [4] as well as commercial ventures such as Flightaware [5]. Figure 1(a) shows a generated spectrum map using spectrum sensors; note that irregular pattern of received power and also interference-free zone for opportunistic spectrum access. Moreover, the cost of cognitive radios such as [6] has fallen recently. There has been even cheaper (<25$) small form-factor devices such as RTL-SDR dongle [7] which can be attached to a smartphone and used to collect spectrum data as the users roam. Figure 1(b) shows commodity devices and a mobile app used for spectrum sensing,

In the context of above described generation of spectrum occupancy map via sensing data from crowdsourced spectrum sensors, we address two specific challenges in this paper: (i) sparse density of spectrum sensors at any instant, (ii) dynamic (i.e., temporally changing) spectrum map, e.g., due to change in transmit powers or powering on/off of fixed PUs. See Figure 2. Construction of spatio-temporal spectrum maps from sparse crowdsourced (and hence, mobile) spectrum sensors boils down to the problem of clustering high-dimensional vectors, with each vectors having only a small number of dimensions with known values. We solve this clustering problem efficiently by leveraging a large amount of historical data— which is practical in our context, due to the crowdsourced nature of the data collection model. We propose a clustering method that first interpolates some of the unknown values in each vector, and then "merges" vectors based on appropriately defined correlations. Via extensive simulations, we demonstrate that our simple approach outperforms other potential approaches.

The rest of the work is organized as follows. We formally define used terms and our problem in Section §III, and present our algorithm for constructing spatio-temporal spectrum maps in Section §IV. We present our performance results in Section §V, and present concluding remarks in Section §VI.

II. Related Works

A large number of recent works [8]–[14] have addressed creating a single spectrum map from sparse sensor measurements via interpolation (often using Ordinary Kriging) techniques. In particular, [4], [15] uses Kriging-based approaches and [16] uses Kriged Kalman Filter to generate spectrum maps. Various machine learning techniques have also been used to generate spectrum maps from sparse sensors [17]–[21]. However, as mentioned above, these works implicitly assume a temporally static spectrum map and thus, the problem addressed is different than our problem of creating multiple spectrum maps in the context of dynamically changing spectrum maps. Authors in [22] do consider a temporally changing spectrum map, but considers a different problem of establishing an upper-bound on difference between known and unknown spatio-temporal power values. To the best of our knowledge, the only work that considers the dynamic scenario as ours is [23] which uses Hidden Markov Models (HMM) to classify the measurement data depending on the combined state of each PU which can be either active or idle. In essence, they assume that the temporal sequence of the on/off states of the PUs has a "pattern" and exploit this pattern via an HMM based technique. In contrast, our setting and techniques do not assume any pattern in the temporal sequence of the configurations.

In other works, spectrum occupancy detection method is explored in [24], [25] with a comprehensive survey on spectrum sensing and related issues in [26]. Large scale spectrum monitoring using high-end sensors is presented in [27]–[30]. Mobile-based crowdsourcing applications for spectrum monitoring are considered in [9], [31], [32].

III. Spectrum Map Generation Problem

In this section, we define some preliminary terms, and formally define the problem addressed in the paper. For simplicity, throughout this paper, we assume that all PUs transmit in the same channel or frequency band; different channels can be handle independently.

PU State, and Configuration. As suggested in the previous section, a primary user (PU) is a licensed incumbent user. At any instant of time, each primary user has a *state* characterized by certain parameters that influences its received signal, e.g., location, transmit power, antenna orientation, etc. For most primary users which are fixed at a location, the state will be generally characterised by its transmit power. Now, consider a set of n PUs in a given region. At any instant of time, the *configuration* of the system is defined as the combination of the states of the PUs in that instant. Thus, if each PU p_i is in a state s_i at a given instant, then the configuration of the system at that instant is $(s_1, s_2, \ldots, s_n)$. In general, if each PU p_i can be in q_i different states, then the total number of possible configurations of the system is $q_1 \times q_2 \ldots \times q_n$. However, in reality, only a subset of these potential configurations may manifest or be observed. We use $\mathcal{C}$ to denote the set of observed configurations. Note that each observed configuration will have a different spectrum map, as described below.

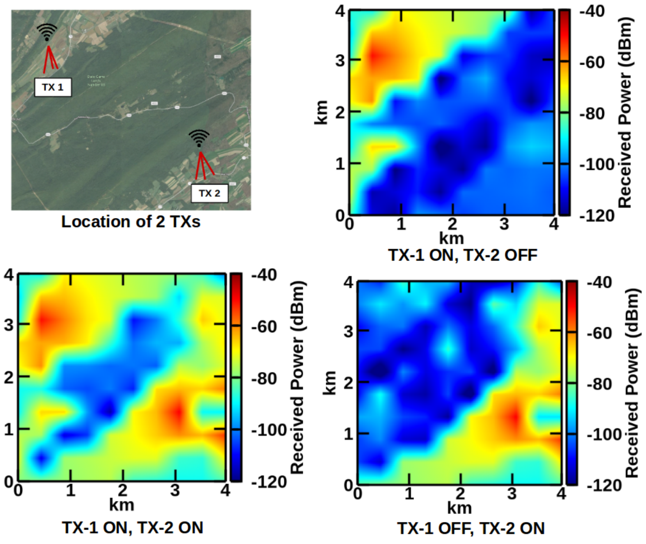

Fig. 2. Spectrum maps generated for ON/OFF states of two nearby TXs

Observation Vectors. Consider a given area A. To limit the number of locations of interest, we divide the given area A into a grid of cells, and concern ourselves with only one spectrum power value/distribution per cell. The goal of our work is to develop techniques to create spectrum maps (formally defined below) for each of the observed configurations with sufficient accuracy. The input to this problem is a set of "observation vectors", with each observation vector being a vector of sensed spectrum powers at various locations at a given instant (based on the configuration active at that instant). More formally, an *observation vector* at a given instant is a vector $(o_1, o_2, \ldots, o_k)$ of sensed spectrum powers, where o_i is either an observed value at i^{th} cell or an unknown value (if there no spectrum sensor was present at any location in the i^{th} cell). We assume that observed value at any location is an aggregate of the signal powers from various PUs in the region, and is perturbed by a zero-mean random noise (thus, the power value at a location may not be same across multiple instants, even if the system is at the same configuration across these instants).

Spectrum Map. For a given area and a configuration $c \in \mathcal{C}$, the *spectrum map* denoted by M_c is a vector $(d_1, d_2, \ldots, d_k)$ where k is the number of cells (locations of interest) in the area, and d_i denotes the parameters (mean and variance) of the received power distribution in the i^{th} cell, in the given configuration c. The set of all spectrum maps, one for each observed configuration, is denoted by $\mathcal{M} = \{M_c | c \in \mathcal{C}\}$.

Creation of Dynamic Spectrum Maps (CDSM) Problem. Given an area with a set of observable configuration $\mathcal{C}$, and a large number of observation vectors, the CDSM problem is to create a set of spectrum maps $\mathcal{M}$ with a spectrum map for each of observed configuration $c \in \mathcal{C}$, such that the overall "accuracy" (e.g., with respect to the "ground truth") of the created spectrum maps is maximized. We discuss appropriate accuracy measures, in the subsection below.

As an example, Figure 2 shows three spectrum maps generated by the ON/OFF states of two PUs. Signal reception at each point in the 4km×4km area is affected by both the PUs. A hilly region between the PUs is reflected in the irregular pattern of the received powers in the 3 maps. Given a large number of crowdsensed observation vectors over time, the CDSM problem is to construct each of the 3 maps as accurately as possible.

A. Performance Metrics

We propose to use two metrics for measurement of accuracy of created spectrum maps: (i) Average Error (AE) when the "ground truth" or true spectrum maps (i.e., the actual spectrum maps based on the PUs' parameters and propagation model, or observed power distribution at each cell) are known, as is the case of simulations or controlled experiments; (ii) Prediction Error (PE) [33] when the true spectrum maps are not unavailable. In addition, we use Adjusted Rand Index (ARI) [34] to measure the *quality of clustering* of all the methods presented here. We now briefly describe these metrics.

Average Error (AE). Consider a given set of k true spectrum maps, and a set of l spectrum maps output by a CDSM algorithm, we define the Average Error (AE) metric by associating each output spectrum map with one of the true spectrum maps, computing the errors for each of these associations, and adding up the errors. Note that l may not be equal to k, and hence, creating the associations between the two sets is not obvious. We propose the following method to create associations. We create a bipartite graph between the two sets of spectrum maps, with true spectrum maps forming one set of nodes and output spectrum map forming the other set. We can represent the associations via an edge between the nodes, one from each set. Each edge between nodes i and j is given a weight $w(i,j)$ equal to the "error" between the two corresponding spectrum maps, as described below. To create most effective associations, we (i) first, run a greedy minimum-cost matching algorithm on the above bipartite graph to find a set of associations that covers all the maps of the smaller set, and (ii) then, if the set of output spectrum maps is larger than the set of true spectrum maps, run a greedy minimum-cost algorithm to create additional associations/edges to cover the remaining output spectrum maps. The "average error" (AE) is then computed by by averaging the weight of all the edges/associations created above. That is,

$$AE = \frac{\sum_{\forall (i,j) \in S} w(i,j)}{|S|} \tag{1}$$

where S is the set of created associations in the greedy algorithm above and $w(i,j))$ is the error between the two maps. The error $w(i,j)$ between two spectrum maps is defined as:

$$w(i,j) = \frac{\sum_{m=1}^{d} |v_m(i) - v_m(j)|}{\sum_{m=1}^{d} |v_m(i)|} \tag{2}$$

where d is the total number of cells, $v_m(i)$ is the power value at cell m in the i^{th} map.

Prediction Error (PE). If the true spectrum maps are not known, then we can compute the "prediction" error of the output spectrum maps as follows [33]. For each observed vector $\mathbf{o}$=$[o_1, \ldots, o_i, \ldots, o_d]$, we pick a random dimension i with known value and use it eventually computing the prediction error; only the remaining (i.e., other than i) known values of o are the used by a CDSM algorithm to generate the output spectrum maps. Let W be the set of values (one from each observation vector) kept for computing the eventual prediction error. Once the output spectrum maps have been created, we compute the overall prediction error PE:

$$PE = \frac{1}{|W|} \sum_{\forall o_i \in W} \frac{|o_i - \hat{o_i}|}{o_i} \tag{3}$$

Above, $\hat{o_i}$ is the predicted value of the original value o_i, computing using the spectrum map M_c to which the vector o is clustered into by the CDSM algorithm.

Adjusted Rand Index (ARI). Let n be the total number of observation vectors to be clustered. Then, for a clustering solution, the Rand Index (RI) is defined as $m/{}^nC_2$, where m is the number of *pairs* of observation vectors a and b such that they are in the same (or different) configuration(s) in the ground truth as well as the output clusters. The value of RI can be anywhere between 0 (worst) and 1 (perfect). The *Adjusted Rand Index* (ARI) is the normalized version of RI, defined as:

$$ARI = \frac{\text{RI - ExpectedRI}}{\text{MaximumRI - ExpectedRI}} \tag{4}$$

The value of ARI can varying between -1 and 1, with a positive value signifying better than random clustering, and a negative value signifying a worse than random clustering. A value close to 1 is regarded as good clustering performance.

IV. Our Approach

Any potential solution to the CDSM problem essentially involves clustering the high-dimensional observation vectors, with one cluster for each configuration and its spectrum map. There are many well-known techniques for clustering such as K-Means, Expectation-Maximization (EM) method etc. [34]. However, the unique challenges in our context is that the given observation vectors might have very few known values, due to sparse availability of spectrum sensors in a large area; using interpolation techniques to estimate the missing values for a vector will most certainly introduce prohibitive estimation errors. In addition, the observation values are noisy, and the number of spectrum maps, i.e., configurations, is generally unknown. In the below subsection, we describe a correlation-based clustering method based on merging of similar vectors. In the following subsection, we describe modified K-Means and EM methods adapted to our context.

A. Correlation-based Merging (CBM) Method

The *correlation-based merging (CBM)* merges two vectors at a time based on their similarity to form a representative clustered vector which in turn is iteratively merged with other vectors. This repeated merging is continued until all the given observation vectors have been merged, and a terminating condition satisfied. To facilitate the merging process, in each vector, we interpolate the unknown values that are spatially close to the known values; here, by spatially we mean whose corresponding locations/cells are geographically close. We use only the spatially close known values for interpolation purposes. Second, the terminating condition obviates the need to know the number of desired clusters. Below, we describe the three key details in the above described Correlation Based Merging (CBM) method: (i) a *diffusion* method to interpolate unknown values and determine correlation between vectors with otherwise few common known values, (ii) a *distance* or dissimilarity metric between the diffused observation vectors, and (iii) the *merging* process, and (iv) the threshold and terminating condition.

Diffusion. Since the observation vectors are gathered from crowdsensed geographically distributed spectrum sensors, the known values within an observation vectors can be assumed to be randomly distributed among the vector dimensions, due to the random motion of the participating crowd. In addition, each vector has very few known values. Thus, for any two observation vectors, it is very likely that they would have very few, if any, common dimensions with known values. This makes determination of putting them in the same cluster (and thus, "merging" them) challenging. However, as spatial correlation exists within in close vicinity for observed received power values, we can use interpolation techniques to estimate unknown values and thus, increase the likelihood of common dimensions between a pair of observations vectors. In particular, interpolation methods such as Inverse Distance Weighting (IDW), K-nearest, Original Kriging [4], [12] are commonly used in the context of spectrum maps. We term this interpolation step as "diffusion". In particular, we interpolate any location l that is $\leq \Delta d$ distance away from one or more locations with known values. We use all the locations that are at most Δd distance away in the interpolation such as IDW. We find that $\Delta d \leq 4$ grids are sufficient for merging datapoints in clustering. We present the sensitivity analysis and the interpolation scheme in Section V.

Distance Metric. After each value has been diffused for a given distance Δd, we find "distance" metric $s(i, j)$ for a pair of observation vectors h_i and h_j as follows.

$$\boxed{s(i,j) = \sum_{l \in L_{ij}} \left(\frac{w_i(l) + w_j(l)}{\sum_{l \in L_{ij}} (w_i(l) + w_j(l))} \right) \times |v_i(l) - v_j(l)|}$$

Here, L_{ij} is a set of locations that have values (either known or diffused) for vectors h_i and h_j, $v_i(l)$ is the received power (dBm) at location l in vector h_i and $w_i(l)$ is a "weight" metric for location l in vector h_i that is defined as follows: if location l has a known value, then $w_i(l)$ is inverse of a small constant c such as 0.001; if l has a diffused value, then $w_i(l)$ is inverse of the distance to the closest location with a known value used to interpolate its value. The intuition is that, the difference in the values at location l i.e. $|v_i(l) - v_j(l)|$ is weighted by the "quality" of the location values in the two vectors. E.g., if both vectors have known value for the location, then the weight is highest $(2/c)$, but if one of the vectors has diffused value from a location with known value that is d_t (minimum) distance away, then the weight is $(1/c + 1/d_t)$. Thus, both the difference in values and the quality of the values contribute to the above defined distance metric. Finally, the weights are normalized by dividing by the sum of all such weights for a set of locations L_{ij}. In this way, we can derive distance metric for clustering for extremely sparse datapoints without introducing significant interpolation error in the pre-processing step of clustering.

Merging. Once distance metric has been calculated for each pair of vectors, we start clustering the vectors by iteratively merging the pair of vectors with minimum distance metric value. In particular, vectors h_i and h_j are merged to form a new merged vector h_{ij} as follows: each location $l \in L_{ij}$ with known values are copied into h_{ij}. If l has known values for both the vectors, both the known values are saved in a list[1]. Then, the diffusion step is re-run for this new vector h_{ij}. The vector h_{ij} replaces both h_i and h_j in the given set of observation vectors. Distance metric $s((i,j),k)$ is calculated between h_{ij} and the rest of the vectors $h_k \in \mathcal{H} - \{h_i, h_j\}$ and the merging process is repeated for the new set of observation vectors.

Threshold and Termination. One important factor in termination of the above iterative merging is a threshold distance s_{max}. Two vectors are only merged if the distance metric between them $\leq s_{max}$. This threshold s_{max} has a bearing on the number of output clusters—if the threshold is too small, then the number of resulting clusters or maps may be too many, and vice-versa. We also note that, initially, the datapoints are extremely sparse, therefore the distance between the vectors is relatively high due to much contribution from the diffused values; however, as the merging progresses, the resulting

[1]During merging process, the list for a location may grow with known values. We assume a Gaussian distribution of known values in presence of sensor noise. Therefore, a Gaussian fit is derived for such a set of known values and the resulting mean is used for subsequent diffusion or interpolation.

vectors have more and more known values resulting in lower and lower distance with other vectors. To incorporate the above progression of distance metric, we tie the threshold s_{max} with the total number of known values in the vectors under consideration. In particular, when comparing vectors h_i and h_j with k_i and k_j number of known values respectively, we set the threshold for their merging as $\theta(i,j) = s_{max}\left(1 - e^{\frac{k_i+k_j}{M}}\right)$ where N is the total number of known values in all the originally given observation vectors. This formulation ensures that the threshold approaches s_{max} as the merging process progresses. In our simulations, we used the standard deviation of all known values in the history as the s_{max} value.

Algorithm 1: Correlation-based Merging (CBM) Method

```
Data: Set of observation vectors $\mathcal{H}$, diffusion parameter $\Delta d$
Result: Set of Clusters or Spectrum Maps $\mathcal{M}$
repeat
    Diffuse $\forall h_i \in \mathcal{H}$ by $\Delta d$;
    Calculate Distance $s(i,j)$ and threshold $\theta(i,j)\ \forall h_i,\ h_j \in \mathcal{H}$;
    Find $min(s(i,j))$ s.t. $s(i,j) \leq \theta(i,j)$;
    Merge $h_i$ and $h_j$ into $h_{ij}$;
    Update $\mathcal{H} \leftarrow \mathcal{H} \cup \{h_{ij}\} - \{h_i, h_j\}$;
until $s(i,j) > \theta(i,j)\ \forall h_i,\ h_j \in \mathcal{H}$;
```

Time Complexity. The number of iterations can be up to $|\mathcal{H}|$, and the most expensive step within each iteration is the **Calculate** step which can take $O(|\mathcal{H}|^2 d)$ time in worst case, where d is the dimension of each vector. Thus, the worst-case time complexity of Algorithm 1 is $O(|\mathcal{H}|^3 d)$.

B. Adapting Known Clustering Approaches

Here, we present two other clustering approaches and their adaptation to our context. We choose K-Means (KM) clustering for its simplicity and suitability for high-dimensional clustering [34]. Also, we can consider the transmitter states as "latent" variables and the resulting spectrum maps as outcomes. As a result, it is natural to use Expectation Maximization (EM) algorithm for clustering [34]. In the following, we present the two algorithms and describe the way the missing values in the input vectors are handled in each case. For both the cases, the number of clusters i.e spectrum maps is derived from our method first.

K-Means Algorithm.

1) **Initialization Step**: Determine the missing values in each d-dimensional input vector using interpolation. Randomly assign the vectors to one of the maps and initialize the spectrum maps as the mean of the vectors.
2) **Clustering Step** Run K-Means clustering [34] using the distance metric $e(i,j)$ between vector i and map j as defined in Equation 2.

EM Algorithm. EM algorithm has two interlocking steps: (i) E-step: the value of the map-membership (latent variable) of the input vectors are estimated using parameters such as mean and variance of values at each dimension; (ii) M-step: vectors that are clustered probabilistically in the E-Step, are used to update the parameter values. These two steps are repeated until the convergence is reached or the marginal improvement is insignificant. We adopt the EM algorithm to handle missing values in the input vectors as follows:

1) **Initialization:** Determine the missing values in each d-dimensional input vector $\mathbf{o_t}$ using interpolation. Randomly choose mean μ and variance σ^2 for each dimension of each map M_c.
2) **E-step:** For each vector $\mathbf{o_t}$ and each map M_c, using the parameters μ and σ^2, find the likelihood $l_{t,c}$ for $\mathbf{o_t} \in M_c$.
3) **M-step:** For each map M_c, update the parameter values μ and σ^2 using the likelihood values $l_{t,c}$
4) **I-step:** Interpolate each missing value v_m at dimension m in vector $\mathbf{o_t}$ as follows:
$$v_m = \frac{\sum_{j \in S} (l_{j,c} \times v_m(j))}{\sum_{j \in S} l_{j,c}} \qquad (5)$$
where S is the set of vectors that belong to the same map M_c as $\mathbf{o_t}$ and each vector $\mathbf{o_j} \in S$ has a known value $v_m(j)$ at dimension m.
5) Go back to Step 2 until convergence.

Note added Step 4 (Interpolation or I-step) to handle extremely sparse input vectors.

V. RESULTS

In this section, we compare the performance of the three methods described both by simulation (with ground truth) and using real data (without ground truth).

Simulation. Here, we generate the input vectors by simulation. As such, the ground truth or the true map-membership of each input vector is available for computation of AE metric.

Experiment Setting. We used data generated from a terrain-based wireless pathloss analyzer tool SPLAT! [35] and used terrain database of US Geological Survey [36]. We put certain number of transmitters in an area of 4km×4km and tuned their Effective Radiated Power (ERP) accordingly such that the transmitters do not interfere with each other. We considered 800MHz frequency band and at most 100 feet antenna height. We divided the area into 100×100 grid. The receiver height was constant at 5 feet over the terrain surface. We created different configurations by turning these transmitters on and off, changing the ERP and both. We than sub-sampled the resulting received power maps to created sparse datapoints. We then added zero-mean Gaussian noise with variance up to 10dB to simulate sensor noise.

Observation. As shown in Figure 3, our method Correlation-based Merging (CBM) is compared with K-Means (KM) and EM-algorithm (EM) for varying (a) known dimensions, (b) configurations (c) sensor noise levels and (d) number of input vectors. The suffixes "AE" and "ARI" indicate the error and the clustering metric respectively. Here, the default parameter values are as follows: %known values = 0.5, # of configuration

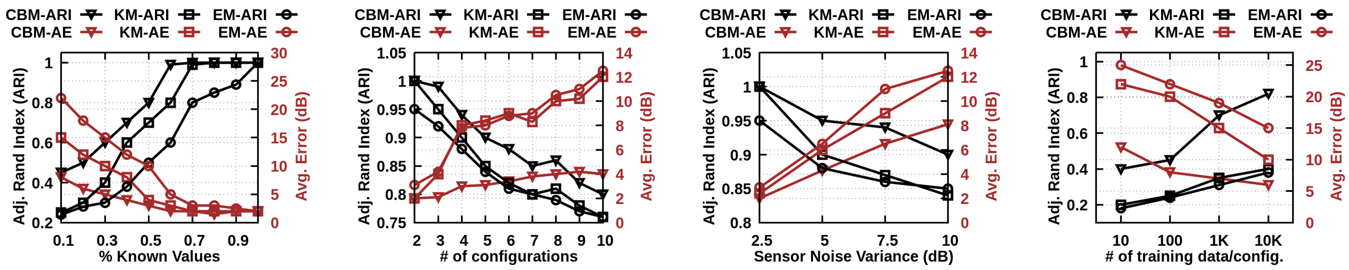

Fig. 3. Performance Comparison for simulation data for varying (a) % known values (b) # of configurations (c) Sensor noise variance (d) # of training data

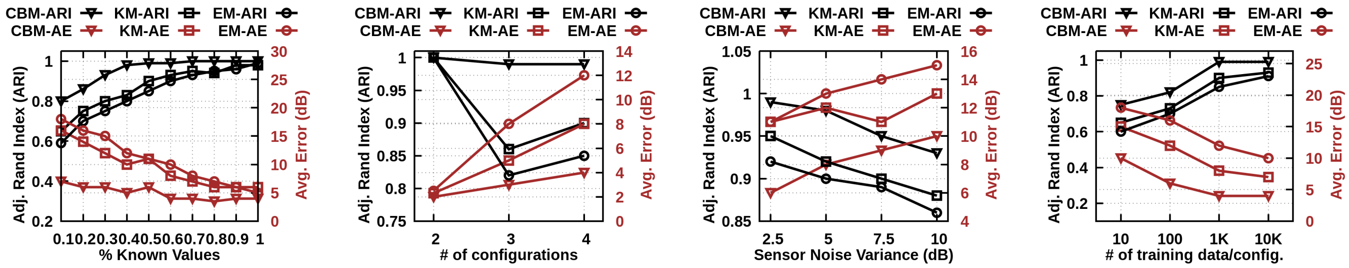

Fig. 4. Performance Comparison for WiFi data for varying (a) % known values (b) # of configurations (c) Sensor noise variance (d) # of training data

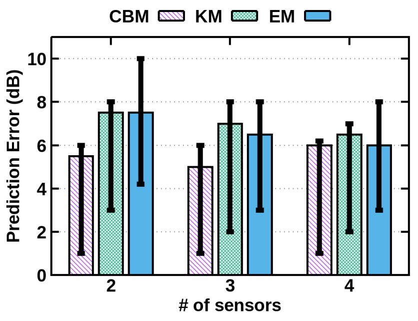

Fig. 5. Performance Comparison for LTE data for varying # of sensors

= 6 and noise variance = 2.5dB and # of training data = 1K. We observe that CBM outperforms both KM and EM algorithms in all cases. In particular, For extremely sparse datapoints such as 0.1% of full dimension, the gap both in AE and ARI metrics are significant. Also, this observation holds for varying number of configurations, sensor noise levels and input vectors. We note that the gap between performance decreases for call cases with the increase in input size, as expected.

Real Data. Here, we apply the methods on real data collected by commodity sensors. We use two dataset as described below.

WiFi Data. We set two WiFi transmitters in the outdoor and collect received powers using commodity sensors shown in Figure 1 equipped with GPS. We turn the two WiFi transmitters on and off to create a total of 4 configurations and collect data with high spatial granularity for each case. We then subsample the data to create input vectors with required % known values for the dimensions. As shown in Figure 4, CBM outperforms the rest in all metrics, even though, the other two perform better compared to simulated case, due to small number of configurations.

Cellular Data. Using the commodity sensors, we collected received powers for the LTE downlink at 751MHz band (2MHz bandwidth) of a popular cellular company. Samples were collected in 1500 locations spanning indoors and outdoors covering a an area of about 15K sq-meters. Since the ground truth regarding the transmitters states are unknown, we use prediction error (PE) as performance metric. Our algorithm outputs two spectrum maps. We also observed from the time-stamped data that, one map occurs during 8am to 12am, the other one for the rest of the time. As shown in Figure 5, CBM outperforms the rest in this metric. With the increase in sensor density, the performance gap diminishes. However, using only 2 sensors, we are able to construct spatio-temporal spectrum maps with $\leq$ 5dB error margin, which is the typical noise variance of the sensors we used.

VI. Conclusion

In this work, we presented a novel clustering method for generating spectrum maps from spatio-temporal data. In particular, we addressed both the time-varying spectrum maps and sparse datapoints. We adopted diffusion method and distance metric for a correlation-based clustering. We show that our method result in high-accuracy maps without requiring any information regarding the transmitters, propagation model, environment etc. In future, we plan to explore a robust prediction mechanism based on spatio-temporal datapoints.

VII. Acknowledgement

This work was supported by NSF awards #1642965.

References

[1] J. Andrews *et al.*, "What will 5G be?" *IEEE JSAC*, 2014.
[2] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[3] Federal Communications Commission, “3.5 GHz Band / Citizens Broadband Radio Service,” https://tinyurl.com/y9krqbvr.

[4] A. Chakraborty, M. S. Rahman, H. Gupta, and S. R. Das, “SpecSense: Crowdsensing for efficient querying of spectrum occupancy,” in *IEEE INFOCOM*, 2017, pp. 1–9.

[5] “FlightFeeder for Android, FlightAware,” http://flightaware.com/adsb/android/.

[6] Ettus Research, “USRP Bus Series,” https://tinyurl.com/nqb67oh.

[7] RTL-SDR Project, “RTL2832U,” https://www.rtl-sdr.com/.

[8] A. Achtzehn, J. Riihijarvi, and P. Mahonen, “Improving accuracy for TVWS geolocation databases: Results from measurement-driven estimation approaches.” in *Proc. IEEE DySPAN*, 2014. [Online]. Available: http://www.inets.rwth-aachen.de/fileadmin/templates/images/PublicationPdfs/2014/DySPAN-2014-TVWS-Estimation.pdf

[9] A. Achtzehn, J. Riihijärvi, I. A. Barriía Castillo, M. Petrova, and P. Mähönen, “CrowdREM: Harnessing the power of the mobile crowd for flexible wireless network monitoring,” in *Proc. ACM HotMobile*, 2015.

[10] M. Molinari, M. Fida, M. K. Marina, and A. Pescapè, “Spatial interpolation based cellular coverage prediction with crowdsourced measurements,” in *Proc. SIGCOMM Workshop on C2B(I)D*, 2015.

[11] C. Phillips, M. Ton, D. Sicker, and D. Grunwald, “Practical radio environment mapping with geostatistics,” in *Proc. IEEE DySPAN*, 2012. [Online]. Available: http://www.eecs.berkeley.edu/ sahai/Papers/DySpAN09.WhitespaceCapacity.pdf

[12] X. Ying, S. Roy, and R. Poovendran, “Incentivizing crowdsourcing for radio environment mapping with statistical interpolation,” in *Proc. IEEE DySPAN 2015*, 2015. [Online]. Available: http://dx.doi.org/10.1109/DySPAN.2015.7343932

[13] X. Ying, C. W. Kim, and S. Roy, “Revisiting TV coverage estimation with measurement-based statistical interpolation,” in *Proc. IEEE COMSNETS*, 2015.

[14] V. Atanasovski, J. van de Beek, A. Dejonghe, D. Denkovski, L. Gavrilovska, S. Grimoud, P. Mhnen, M. Pavloski, V. Rakovic, J. Riihijarvi, and B. Sayrac, “Constructing radio environment maps with heterogeneous spectrum sensors,” in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2011, pp. 660–661.

[15] Y. Hu and R. Zhang, “Secure crowdsourced radio environment map construction,” in *IEEE ICNP*, 2017.

[16] S. Kim, E. Dall’Anese, and G. B. Giannakis, “Cooperative spectrum sensing for cognitive radios using kriged kalman filtering,” *Journal on Selected Topics in Signal Processing*, vol. 5, 2011.

[17] J. A. Bazerque and G. B. Giannakis, “Distributed spectrum sensing for cognitive radio networks by exploiting sparsity,” *IEEE Transaction on Signal Processing*, vol. 58, 2010.

[18] J. A. Bazerque, G. Mateos, and G. B. Giannakis, “Group-lasso on splines for spectrum cartography,” *IEEE Transaction on Signal Processing*, vol. 59, 2011.

[19] J. A. Bazerque and G. B. Giannakis, “Nonparametric basis pursuit via sparse kernel-based learning: A unifying view with advances in blind methods,” *IEEE Signal Processing Magazine*, vol. 30, 2013.

[20] B. A. Jayawickrama, E. Dutkiewicz, I. Oppermann, G. Fang, and J. Ding, “Improved performance of spectrum cartography based on compressive sensing in cognitive radio networks,” in *IEEE ICC*, 2013.

[21] S. Kim and G. B. Giannakis, “Cognitive radio spectrum prediction using dictionary learning,” in *IEEE GLOBECOM*, 2013.

[22] A. Ahuja, V. J. Ribeiro, R. Chandra, and A. Kumar, “SpectraMap: Efficiently Constructing a Spatio-temporal RF Spectrum Occupancy Map,” in *Communication Systems and Networks*. Springer International Publishing, 2017, pp. 53–71.

[23] K. Ichikawa and T. Fujii, “Radio environment map construction using hidden markov model in multiple primary user environment,” in *2017 International Conference on Computing, Networking and Communications (ICNC)*, 2017.

[24] F. F. Digham, M. S. Alouini, and M. K. Simon, “On the energy detection of unknown signals over fading channels,” *IEEE Transactions on Communications*, vol. 55, pp. 21–24, 2007.

[25] K. Kim, I. A. Akbar, K. K. Bae, J. S. Um, C. M. Spooner, and J. H. Reed, “Cyclostationary approaches to signal detection and classification in cognitive radio,” in *IEEE DySPAN*, 2007, pp. 212–215.

[26] F. Hu, B. Chen, and K. Zhu, “Full spectrum sharing in cognitive radio networks toward 5g: A survey,” *IEEE Access*, vol. 6, pp. 15 754–15 776, 2018.

[27] “Microsoft Spectrum Observatory project,” https://observatory.microsoftspectrum.com/.

[28] A. Iyer, K. K. Chintalapudi, V. Navda, R. Ramjee, V. Padmanabhan, and C. Murthy, “SpecNet: Spectrum sensing sans frontieres.” in *Proc. NSDI*, 2011. [Online]. Available: http://research.microsoft.com/pubs/142837/SpecNet-NSDI11.pdf

[29] J. Naganawa, H. Kim, S. Saruwatari, H. Onaga, and H. Morikawa, “Distributed spectrum sensing utilizing heterogeneous wireless devices and measurement equipment,” in *Proc. IEEE DySPAN*, 2011.

[30] T. Zhang and S. Banerjee, “A Vehicle-based Measurement Framework for Enhancing Whitespace Spectrum Databases.” in *Proc. ACM MobiCom*, 2014.

[31] J. Shi, Z. Guan, C. Qiao, T. Melodia, D. Koutsonikolas, and G. Challen, “Crowdsourcing access network spectrum allocation using smartphones,” in *Proc. ACM HotNets*, 2014.

[32] D.-H. Shin, S. He, and J. Zhang., “Joint sensing task and subband allocation for large-scale spectrum profiling,” in *Proc. IEEE INFOCOM*, 2015.

[33] A. Rencher and W. Christensen, *Methods of Multivariate Analysis*. Wiley, 2012.

[34] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[35] John Magliacane, KD2BD, “ Signal Propagation, Loss, And Terrain (SPLAT) ,” http://www.qsl.net/kd2bd/splat.html.

[36] US Geological Survey, “SRTM,” https://tinyurl.com/zwkpk9h.