IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 24, 2016, accepted April 7, 2016, date of publication April 21, 2016, date of current version May 9, 2016.

Digital Object Identifier 10.1109/ACCESS.2016.2557460

On a Training-Less Solution for Non-Intrusive Appliance Load Monitoring Using Graph Signal Processing

BOCHAO ZHAO, LINA STANKOVIC, (Senior Member, IEEE), AND VLADIMIR STANKOVIC, (Senior Member, IEEE)
Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow G1 1XW, U.K.

Corresponding author: V. Stankovic (vladimir.stankovic@strath.ac.uk)

This work was supported by the U.K. Engineering and Physical Sciences Research Council under the Transforming Energy Demand in Buildings through the Digital Innovation (BuildTEDDI) Funding Programme under Grant REFIT EP/K002368.

ABSTRACT With ongoing large-scale smart energy metering deployments worldwide, disaggregation of a household's total energy consumption down to individual appliances using analytical tools, also known as non-intrusive appliance load monitoring (NALM), has generated increased research interest lately. NALM can deepen energy feedback, support appliance retrofit advice, and support home automation. However, despite the fact that NALM was proposed over 30 years ago, there are still many open challenges with respect to its practicality and effectiveness at low sampling rates. Indeed, the majority of NALM approaches, supervised or unsupervised, require training to build appliance models, and are sensitive to appliance changes in the house, thus requiring regular re-training. In this paper, we tackle this challenge by proposing an NALM approach that does not require any training. The main idea is to build upon the emerging field of graph signal processing to perform adaptive thresholding, signal clustering, and pattern matching. We determine the performance limits of our approach and demonstrate its usefulness in practice. Using two open access datasets—the US REDD data set with active power measurements downsampled to 1 min resolution and the UK REFIT data set with 8-s resolution, we demonstrate the effectiveness of the proposed method for typical smart meter sampling rate, with the state-of-the-art supervised and unsupervised NALM approaches as benchmarks.[1]

INDEX TERMS Non-intrusive appliance load monitoring, load disaggregation, graph signal processing.

I. INTRODUCTION

With large-scale smart energy metering deployments that are ongoing or planned in many countries, there is a growing interest in data analytical research to maximize benefits from the collected data via richer energy feedback, novel energy saving services and more flexible pricing mechanisms that are useful to householders and other stakeholders.

Load disaggregation [2], that is, identifying the individual appliance load from the total, aggregate load, can deepen energy feedback leading to more efficient use of appliances (up to 20% of reduction in energy consumption is expected via appliance-feedback and specific appliance upgrade programs [3]). It can support home automation and appliance upgrade decisions, as well as enable activity recognition [4]. Energy suppliers can better forecast demand, system operators can monitor the effect of smart grid fluctuations on the residential microgrid, and appliance manufacturers can optimise product design to meet customer usage habits.

Non-intrusive appliance load monitoring (NALM) [2] is an attractive method that disaggregates the total load from an electricity meter purely using data analytics (software tools), that is, without sub-metering or additional hardware. For the average householder to reap the benefits of smart metering, NALM should ideally be designed to operate purely on measurements that are already collected for metering and billing purposes. Hence, techniques that work with active power only at resolutions of seconds and minutes [5] are of special interest and could be used with a wide range of current off-the-shelf smart metering devices, shifting research

[1] Part of this work was presented at *IEEE GlobalSIP-2015* [1]. The REFIT dataset used to generate the results can be accessed via DOI 10.15129/31da3ece-f902-4e95-a093-e0a9536983c4.

priorities from high-rate approaches that use both voltage and current signatures, to low-complexity, low-rate methods that operate only on the measurements used for billing purposes.

Driven by the host of emerging applications, in this paper, we propose a new, *blind*, low-rate NALM approach that does not require any training. The proposed approach disaggregates any aggregate active power dataset without any prior knowledge, including knowldge of appliances contributing to the aggregate or their number. It relies on graph signal processing (GSP) [6], an emerging field based on representing a dataset using a discrete signal indexed by nodes of a graph. GSP offers an alternative to conventional signal processing approaches by embedding the structure of signals onto a graph, leading to a powerful scalable and flexible approach suitable for a range of applications (see [6]–[9] and references therein). We note that recently, the latter two authors of this paper proposed a GSP-based NALM approach in [10]. However, the approach of [10] is supervised and employs GSP for data classification only.

In contrast to traditional machine-learning approaches, such as Hidden Markov Model (HMM), that require plenty of observations to construct a graph, the proposed graph signal processing approach takes an intuitive approach in constructing a graph without relying on the signal's statistics [11]. Thus, it is expected that the proposed approach will work well in the absence of a training dataset, unlike traditional HMM-based and other machine learning methods [12]–[16].

Specifically, for the purpose of NALM disaggregation of active power signal, GSP is used three times: first for robust event detection, then to perform clustering, and finally for feature matching. The approach is event-based and relies only on time-series data without any training. We demonstrate good accuracy of the proposed solution using two open access datasets: REDD [17] and REFIT [18].

This paper is organized as follows. Section II briefly reviews pertinent literature on low-rate NALM. Section III reviews the state of the art of GSP. Section IV describes our problem formulation and the proposed disaggregation methodology, followed by analysis of the proposed algorithm and its limits in Section V. Section VI presents a case study. Section VII describes the experimental setup followed by our results and discussion. The research findings are concluded in Section VIII.

II. LOW-RATE NON-INTRUSIVE APPLIANCE LOAD MONITORING (NALM): BACKGROUND AND LITERATURE REVIEW

NALM refers to analytical methods that take as an input electrical parameters (voltage, current, active/reactive power etc) measured at the household's mains meter and output energy consumption, broken down to appliance level. In this paper, we focus on NALM methods that work on active power measurements only, at rates of the order of seconds and minutes, which resembles the type of data, available using smart energy meters deployed on large scale worldwide (see, for example, [5]).

Event-based NALM approaches are based on identifying windows of *events* when substantial statistical change in the power measurement occurs that indicates that one or more appliances have been switched on or off, or change their operational state. After such events are identified (usually via edge detection), features are extracted (e.g., rising/falling power edges, area, time duration) from each such event window, and then classification is performed on the extracted features using a model built during the training process. Different classification methods have been used including support vector machine (SVM), neural networks, and decision trees (see [19]–[23] and references therein). State-based, probabilistic approaches [12]–[14], on the other hand, usually based on HMMs and their variants, rely on building an appliance load model using a finite state machine by learning parameters for prior distributions of appliance states. Similarly, sparse coding based approaches [24], [25] require training data to design appropriate basis functions and dictionaries used to transform the signals based on their sparse nature.

Based on the dependency of the approach on a labelled training set (i.e., a diary of which appliance changed state and when), all NALM approaches can be divided into supervised, semi-supervised, and unsupervised. NALM methods that require a labelled training set, fall in the category of supervised NALM. See [19] and [20] for recent reviews.

However, providing an accurate labelled training set is often impractical due to the requirement for sub-metering or accurate time-diaries. Hence, *unsupervised* approaches have become popular. We group all methods closely related to our work as follows. Group 1 comprises all traditional unsupervised approaches, which require unlabelled training data to build appliance models or populate appliance database. In [26], an overview of unsupervised NALM methods with performance comparison is provided. Group 2 consists of NALM methods that use 'known houses' for building appliance models, which are then used for disaggregation in 'unknown' ('unseen') houses. Methods that do not require training before NALM disaggregation form Group 3.

Group 1 methods are usually based on hierarchical clustering or Hidden Markov Models (HMM)s where appliance models are generated, manually or automatically, during the training phase. The early work on unsupervised NALM is presented in [27] where four low-rate NALM methods are proposed using (conditional) factorial HMM (FHMM) and Hidden semi-Markov models. This method cannot disaggregate base load and refrigerator, and is prone to converge to a local minimum. More recently, new approaches that address some of the weaknesses of [27] are described in [12]–[14] and [28] based on FHMM, differential FHMMs, additive FHMM and Hierarchical Dirichlet Process Hidden Semi-Markov Model factorial structure, respectively. A magnitude-base unsupervised NALM approach presented in [29] uses standard HMM with smoothing to obtain better features. Two FHMM-based approaches in [30] and [31] exploit context information and interactions chains, respectively, to improve performances of standard FHMM.

Although the above probabilistic state-based approaches are unsupervised, they use expert knowledge to set a-priori values for each appliance state, and require a training set (usually where appliance operations do not overlap) to build/refine the state models, such as [16] and [32]. The performance of these methods depends on how well generated models approximate appliance true usage. Thus the longer the training period, the better the results. Moreover, it has not been demonstrated that these methods can be generalized across houses; that is, if applied to a house that was not included in training ('unseen house'), it is uncertain that the methods will perform well. Unsupervised, time-series approaches, such as [23], do not build appliance models, but require training periods to build a database of time-series signatures, necessary for pattern matching. Similarly, the unsupervised approach of [33], based on probabilistic sequential mining and temporal motif mining, requires extra information such as the number of appliances.

NALM methods in Group 2 assume the existence of 'training' or 'known' houses where submetering is done and used to build appliance models or populate an appliance database, which is then used in unknown or unseen houses. In [22], for example, k-means and SVM are combined to disaggregate washing machine in an unknown house using models generated in two 'known' houses. Deep learning is used in [34], where three neural network architectures are adapted to NALM with supervised training in known houses. The method performs well on a house, unseen during training, but as any other deep learning approach, requires a large training set, training is of high computational complexity and does not perform well for multi-state appliances. In [35], it is assumed that if two houses have a similar aggregate consumption during different seasons, it is likely that they will also have a similar consumption at the appliance level. Based on this assumption, instead of performing NALM to disaggregate appliance usage, k-nearest neighbor (k-nn) is suggested in [35] for 'similar' houses, where sub-metering is available, to predict the disaggregated consumption without any NALM.

Though Group 2 approaches do not need training in 'unseen' houses, their main drawback is that they do not work well for uncommon appliances, are sensitive to outliers, require a large set of houses for training where submetering is possible, and cannot generalize well across different geographical locations.

Group 3 approaches are the closest to our method as they do not require training before disaggregation takes place. An unsupervised event detector for NALM presented in [36] applies Kernel Fisher Discriminant Analysis (KFDA) without any training; however, it requires high sampling rates and uses current harmonics. In [37], an unsupervised low-rate NALM approach, based on clustering and matching pursuit is proposed; however, the approach uses both active and reactive power, performs poorly for appliance loads below 400W and concludes that the results might improve only if additional features are included. An unsupervised FHMM-based approach introduced in [38] learns the appliance models on-the-fly, thus its performance gradually improves requiring some time to reach high accuracy.

In this paper, we deviate from Group 1 contributions above, in that our proposed approach does not require any training or any expert/customer input to perform disaggregation, which makes the proposed system practical and potentially massively deployable (see [39] for a discussion about necessary features to ensure NALM practicality). In contrast to Group 2 methods, the proposed method does not require existence of houses with sub-metering.

The proposed method is event-based forming patterns of appliance signatures on-the-fly using clustering, and relies on pattern matching to label the identified patterns. Since our approach is signal processing based (as opposed to machine learning based), unlike the method presented in [38], our approach does not need to learn appliance features over time, since it is not based on the appliance model generation.

III. GRAPH SIGNAL PROCESSING (GSP)

GSP is based on *graph signals* obtained by indexing a dataset by nodes of a graph. The basic idea is to represent a dataset using a graph defined by a set of nodes and a weighted adjacency matrix. Each node in the graph corresponds to an element in the dataset while the adjacency matrix defines all edges in the graph and their weights, where assigned weights reflect the degree of similarity or correlation, between the nodes.

In the following we denote matrices and vectors by bold upper-case and lower-case letters, respectively. For a matrix $\mathbf{A}$, $A_{i,j}$ denotes its entry in the i-the row and j-th column. Similarly, for a vector $\mathbf{x}$, x_i denotes its i-th element. We denote by $\mathbf{x}_{a:b}, a < b$ a vector $[x_a, \ldots, x_b]$ and by $\mathbf{A}_{a_1:b_1,a_2:b_2}$ a sub-matrix of $\mathbf{A}$ containing rows from a_1 to b_1 and columns from a_2 to b_2.

Given an acquired set of measurements $\mathbf{x}$, we define a graph $G = \{\mathbf{V}, \mathbf{A}\}$, where each node $v_i \in \mathbf{V}$ corresponds to one acquired measurement, and edges between the nodes are defined by a weighted adjacency matrix $\mathbf{A}$. The graph signal, $\mathbf{s}$, is then defined as a mapping from a set of nodes $\mathbf{V}$ to a set of complex numbers, where each element s_i is indexed by a node $v_i \in \mathbf{V}$.

The adjacency matrix $\mathbf{A}$ defines all edges in the graph and their weights. The values of $A_{i,j}$ are often naturally defined by the physical meaning of the collected data. If that is not the case, a Gaussian kernel weighting function:

$$A_{i,j} = \exp\left\{-\frac{|dist(x_i,x_j)|^2}{\rho^2}\right\}, \qquad (1)$$

is often used in the literature [8], [10], where ρ is the scaling factor and $dist(x_i, x_j)$ can be, for example, Euclidean distance. The graphs and signals on graphs defined above can be conveniently used to represent very different data structures, such as time series, images, sensors, tracked objects, social networks, hyperlinked documents etc. [6], [7].

A *graph signal's global smoothness* with respect to the intrinsic structure of its underlying graph can be defined as [6]:

$$\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}A_{i,j}\left(s_j-s_i\right)^2, \tag{2}$$

which can be shown to be $\mathbf{s}^T\mathbf{Ls}$, where $\mathbf{L}$ is the graph Laplacian operator [7] given by

$$\mathbf{L}=\mathbf{D}-\mathbf{A}, \tag{3}$$

where $\mathbf{D}$ is a diagonal matrix with nonzero entries $D_{i,i}=\sum_j A_{i,j}$.

If $\mathbf{s}$ is piecewise smooth with respect to underlying graph structure, then $\mathbf{s}^T\mathbf{Ls}$ is generally small. The global graph-signal smoothness can effectively be used as a prior for regularization, since the Laplacian regularizer $\mathbf{s}^T\mathbf{Ls}$ is a good measure of variation in the signal modulated by weights in $\mathbf{A}$. Then, to find the smoothest signal, we can formulate the global smoothness minimization problem as:

$$\arg\min_{\mathbf{s}}\left\|\mathbf{s}^T\mathbf{Ls}\right\|_2^2. \tag{4}$$

If $\mathbf{s}$ is an N-length discrete signal, then $\mathbf{L}$ is an $N\times N$ matrix and [10]:

$$\begin{aligned}\mathbf{s}^T\mathbf{Ls}&=s_1\mathbf{L}_{1,1}s_1+s_1\mathbf{L}_{1,2:N}\mathbf{s}_{2:N}\\&\quad+\mathbf{s}_{2:N}^T\mathbf{L}_{2:N,1}s_1+\mathbf{s}_{2:N}^T\mathbf{L}_{2:N,2:N}\mathbf{s}_{2:N}.\end{aligned} \tag{5}$$

Note that (5) is the same as in [8] and [10], except that we replaced a vector of known samples used for training in the supervised classification approaches [8], [10] with a randomly picked sample, s_1.

Since $\mathbf{D}$ is a diagonal matrix, $\mathbf{L}$ is also diagonally symmetric. Thus, since the first term in (5) does not affect minimization, minimization (4) is simplified as:

$$\begin{aligned}&\arg\min_{\mathbf{s}}\left\|\mathbf{s}^T\mathbf{Ls}\right\|_2^2\\&=\arg\min_{\mathbf{s}_{2:N}}\left\{2\mathbf{s}_{2:N}^T\mathbf{L}_{2:N,1}s_1+\mathbf{s}_{2:N}^T\mathbf{L}_{2:N,2:N}\mathbf{s}_{2:N}\right\}.\end{aligned} \tag{6}$$

As an unconstrained quadratic programming problem, this minimization has a closed form solution [9], [40]:

$$\mathbf{s}^*=\mathbf{L}_{2:N,2:N}^{\#}(-s_1)\mathbf{L}_{1,2:N}^T, \tag{7}$$

where $(.)^{\#}$ denotes the pseudo-inverse matrix. $\mathbf{s}^*$ is the smoothness optimization solution, i.e., a solution that minimizes the total graph variation.

IV. PROPOSED DISAGGREGATION ALGORITHM

In this section we describe the proposed disaggregation algorithm, by first formulating the disaggregation problem and then present an overview of the proposed approach, before describing each of its building blocks.

A. PROBLEM FORMULATION AND NOTATION

Let P_{t_i} be the total household's active power consumption measurement at time instance t_i, for $i=1,\ldots,n$. The task is, for each t_i, to find the power contribution of each individual appliance m, P_{mt_i}, towards the total power consumption P_{t_i}. That is,

$$P_{t_i}=\sum_{m\in\mathcal{M}}P_{mt_i}+n_{t_i}, \tag{8}$$

where $\mathcal{M}$ is the set of known appliances in the house and n_{t_i} is the noise that includes random measurement noise, base load, plus all unknown appliances in the house. Then, we can formulate the disaggregation optimization problem as finding P_{mt_i} such that:

$$\min_{P_{mt_i}}\left|P_{t_i}-\textstyle\sum_{m\in\mathcal{M}}P_{mt_i}\right|. \tag{9}$$

Let $\Delta P_{t_i}=P_{t_{i+1}}-P_{t_i}$, for $i=1,\ldots,n-1$, denote the power variation signal between adjacent aggregate power readings. As the interval $t_{i+1}-t_i$ is constant, we simplify the notation as $\Delta P_{t_i}=\Delta P_i$.

B. ALGORITHM OVERVIEW

We propose an event-based algorithm for finding the solution to the disaggregation problem above. Event-based methods first identify windows of *events*, i.e., statistically significant changes in active power that could indicate that one or more appliances have changed their operational state (for example, switched on/off). Event detection is usually done via edge detection with fixed or adaptive thresholds [23]. After events have been identified, relevant features are extracted from each event window. Finally, the extracted features are classified into pre-defined appliance classes using a model defined during training.

The proposed algorithm follows the above steps, performing data filtering to adapt edge detection thresholds, and clustering to identify events and extract features - active power edges. Then, it replaces the final conventional classification step with a feature matching step.

The proposed method does not require any prior knowledge about the house or appliances therein, such as the number of used appliances and/or their type. In contrast to probabilistic methods that learn the appliance model, offline or online, our signal processing approach does not rely on probabilistic modelling and is not sensitive to adding and removing appliances from the house. Moreover, it performs equally well for frequently used and uncommon appliances. It has been shown to work on aggregate data in the presence of noise and unknown loads, and starts to disaggregate without needing a pre-disaggregation training period.

As is common practice [19], [20], the method works on sliding time windows, whose duration can be adjusted. Window sizes from a month, week down to a day and an hour were tested, but shorter windows are also possible. After each window has been processed, disaggregated appliances are named using a current database, whose contents are updated with new signatures if an appliance is not found. If the database is empty, appliances would be added as they are disaggregated and arbitrary labelled (e.g., Appliance X), until labels are provided, via an app or a web-interface.

Note that, as a training-less unsupervised approach that does not rely on appliance modelling, different power states of multi-state appliances as well as different modes of operation of appliances with different operation cycles, such as air conditioning, are treated as separate appliances. In the final labelling stage, these appliances are labelled as the same appliance using the database.

As with all low-rate NALM algorithms, power states that are magnitude-wise very similar cannot be separated. In Section V, we quantify the bound for which two power states can be separated, depending on load fluctuation and magnitude difference between the power states.

The computational complexity of the proposed algorithm is within both Class ‘P’ in time and class ‘PSPACE’ in space [41], so it is real-time and real-world applicable, and suitable for online applications. The flowchart of the overall method is shown in Fig. 1. The following two subsections detail the two key building blocks in Fig. 1.

C. EDGE DETECTION AND CLUSTERING

The objective of this step is event detection and feature extraction. We define an event as a statistically significant change in ΔP_i that indicates appliance switching on/off or state transitions in multi-state appliances, such as washing machine and air conditioner. Conventionally, events are detected using fixed or adaptive thresholding, where the threshold needs to be large enough to filter out power variations of the same appliance but small enough to capture low loads. In the following, we propose a GSP-based method to find the optimal adaptive threshold for event detection based on the available power readings.

To avoid detecting stand-by settings, we define an initial, small, threshold T_0 (in the order of Watts); thus, all

$$\Delta P_i \in (-\infty, -T_0) \cup (T_0, \infty) \tag{10}$$

will form a set of candidates Π for event detection. Note that $|\Delta P_i| > T_0$ could indicate an event but it could also occur purely due to power load fluctuations (i.e., power load noise). The latter cases need to be filtered out from Π, which we address via iterative GSP-based adaptive thresholding and clustering.

Clustering is performed in two stages. Both stages consist of consecutive passes through the data samples in Π, where in each pass we form a cluster from all data samples most correlated to the first data sample by applying regularization on the underlying graph. In Stage 1 we use the initial threshold T_0 to filter out low-magnitude edges, while in

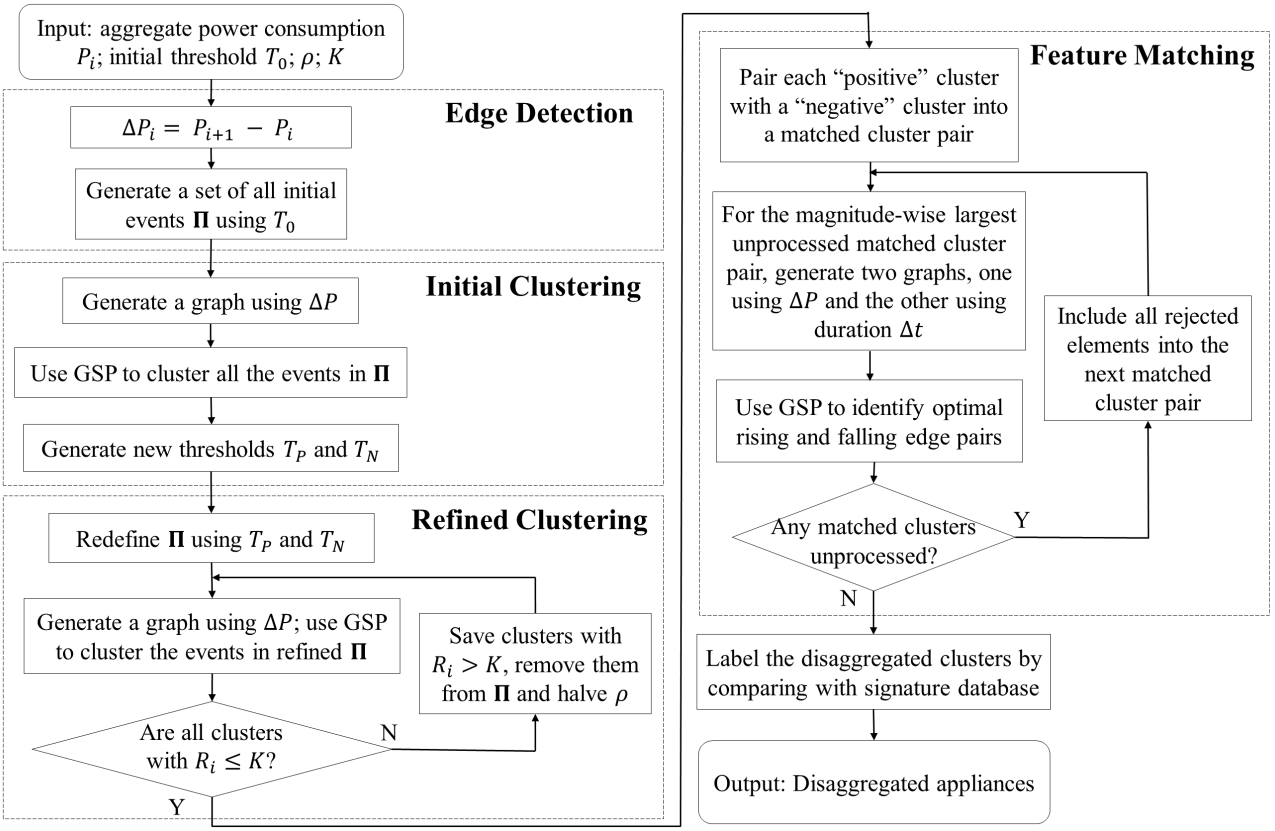

Input: aggregate power consumption P_i; initial threshold T_0; ρ; K
Edge Detection
$\Delta P_i = P_{i+1} - P_i$
Generate a set of all initial events $\mathbf{\Pi}$ using T_0
Initial Clustering
Generate a graph using ΔP
Use GSP to cluster all the events in $\mathbf{\Pi}$
Generate new thresholds T_P and T_N
Refined Clustering
Redefine $\mathbf{\Pi}$ using T_P and T_N
Generate a graph using ΔP; use GSP to cluster the events in refined $\mathbf{\Pi}$
Save clusters with $R_i > K$, remove them from $\mathbf{\Pi}$ and halve ρ
Are all clusters with $R_i \leq K$?
N
Y
Feature Matching
Pair each “positive” cluster with a “negative” cluster into a matched cluster pair
For the magnitude-wise largest unprocessed matched cluster pair, generate two graphs, one using ΔP and the other using duration Δt
Include all rejected elements into the next matched cluster pair
Use GSP to identify optimal rising and falling edge pairs
Any matched clusters unprocessed?
Y
N
Label the disaggregated clusters by comparing with signature database
Output: Disaggregated appliances

FIGURE 1. Flow chart of the proposed algorithm.

Stage 2 we refine some of the clusters by adapting the threshold to the data samples.

Specifically, we start Stage 1 clustering by building a graph using samples in Π, i.e., all samples ΔP_i that are greater than T_0 or less than $-T_0$, associating them to nodes v_i of a graph. $A_{i,j}$ denotes the weight of an edge from node v_i to v_j that depends on the level of correlation between $x_i = \Delta P_i$ and $x_j = \Delta P_j$, calculated as in (1) using the Euclidean distance measure. We set s_1 to 1 if $\Delta P_1 > T_0$ and -1 otherwise, and initialise all $s_j = 0$, for $j > 1$. In the first pass, we cluster all samples statistically similar to s_1.

To do that, we calculate (7). If $s_j^* > q$, where q is a constant fixed through all iterations, then ΔP_j is assigned to the first cluster of events (together with s_1), and is removed from Π. This way, we form the first cluster C_1 of events. In the second pass, we again focus on the first remaining element in Π, and repeat the procedure of generating the graph and calculating (7) using unclustered $\Delta P_i'$s to form a cluster of events statistically similar to the new s_1. Again, only $\Delta P_i'$s for which $s_i^* > q$, will be added to the next cluster of events. We continue with the passes through the remaining data samples, until all candidate events are clustered and Π becomes an empty set.

Note that after Stage 1, each cluster will comprise purely positive or purely negative edges. Let μ_i and σ_i denote, respectively, the mean value and the standard deviation of Cluster C_i. Instead of setting a constant threshold, we evaluate the *quality* of cluster C_i by relative standard deviation (RSD), R_i:

$$R_i = \left|\frac{\sigma_i}{\mu_i}\right|. \tag{11}$$

Note that the lower the R_i, the higher the *quality* of cluster C_i in the sense that the clustered points will be closely grouped together around the mean, indicating good statistical similarity between the cluster elements. If the mean of the cluster is low, the samples will be more prone to noise, thus cluster elements need to be grouped closer together. Therefore, for two clusters with the same variance, the cluster with the larger mean would have better quality.

The mean values of the two clusters with the highest RSD R_i (''worst quality clusters''), among all clusters of positive elements and those of negative elements, will determine T_P and T_N thresholds used for positive and negative edges, respectively. That is, a set of candidate events Π is redefined as:

$$\Pi = \Delta P \in (-\infty, T_N) \cup (T_P, \infty). \tag{12}$$

In the cluster refining stage (Stage 2), we carry out the same GSP-based clustering iterations as above but only re-clustering elements in the low-quality clusters that have $R_i > K$, by halving ρ in (1), or effectively reducing the edge weights for the same correlation, in each pass, where K is a heuristically obtained constant defining the acceptable precision of a cluster. After each pass, all resulting clusters with RSD $R_i \leq K$, will be stored as the final clusters, passed to the following Feature Matching step, and removed from $\mathbf{\Pi}$. The clustering will end when there are no remaining elements in $\mathbf{\Pi}$.

Finally, we merge the smallest-sized clusters (clusters with the least number of elements) into larger-sized clusters to make the number of clusters comprising increasing power edges equal to the number of clusters comprising decreasing power edges.

D. FEATURE MATCHING

Since the final clusters contain ''positive'' clusters (comprising increasing power edges) and the same number of ''negative'' clusters (with decreasing power edges), we pair each ''positive'' cluster with a ''negative'' cluster that has the closest absolute mean value.

Next, for each positive-edge-negative-edge cluster pair, we use GSP to match each element (i.e., increasing power edge) in the positive cluster with an element (decreasing edge) in the paired negative cluster, by exploiting magnitude differences as well as time intervals between the edges, as two matching features.

We start with the magnitude-wise largest cluster, that is, a cluster that has the largest mean. Let $\mathbf{C_P}$ and $\mathbf{C_N}$ denote two paired clusters, that is, the vectors of increasing and decreasing power edges. The task is, for each $C_{Pi} \in \mathbf{C_P}$ to find an optimal match among all candidates $C_{Ni} \in \mathbf{C_N}$. To do this, we form a graph by considering only edges in $\mathbf{C_N}$ that occur after C_{Pi}, since the decreasing edge must occur after the increasing edge, and before C_{Pi+1}. This subset of $\mathbf{C_N}$ will be regarded as a set of candidates, denoted by $\mathbf{\Phi}$. Let $\mathbf{\Phi_M}$ represent the set of magnitude differences between C_{Pi} and each element in $\mathbf{\Phi}$. Let $\mathbf{\Phi_T}$ represent the set time intervals between C_{Pi} and each element in $\mathbf{\Phi}$.

Two graphs are formed:

1) the first graph, $G_M = \{\mathbf{V}_M, \mathbf{A}_M\}$, with nodes indexed by the elements of $\mathbf{\Phi_M}$, and

$$A_{M\,i,j} = \exp\left\{-\frac{|dist(\phi_{M\,i}, \phi_{M\,j})|^2}{\rho^2}\right\}. \tag{13}$$

The graph signal, $\mathbf{s}_M$, is defined as follows: $s_{M\,1}$ is set to be the average value of the elements in $\mathbf{\Phi_M}$, and for $j > 1$, $s_{M\,j} = 0$;

2) the second graph, $G_T = \{\mathbf{V}_T, \mathbf{A}_T\}$, has nodes indexed by the elements of $\mathbf{\Phi_T}$, and

$$A_{T\,i,j} = \exp\left\{-\frac{|dist(\phi_{T\,i}, \phi_{T\,j})|^2}{\rho^2}\right\}. \tag{14}$$

The graph signal, $\mathbf{s}_T$, is defined as follows: $s_{T\,1}$ is set to be the median value of the elements in $\mathbf{\Phi_T}$, instead of the mean value to reduce the influence of the outliers, and for $j > 1$, $s_{T\,j} = 0$;

We compute the graph signal that minimizes the global smoothness of each of the two graphs using (7), obtaining the solutions $\mathbf{s}_M^*$ and $\mathbf{s}_T^*$.

Next we formulate an optimization problem for feature matching, $i = 1, \ldots, n$, as:

$$\arg\max_{i} \{\alpha s_{M_i}^* + \beta s_{T_i}^*\}, \tag{15}$$

where n is the number of candidates, that is, the length of $\mathbf{s}_M^*$ and $\mathbf{s}_T^*$, and α and β are heuristically chosen, with $(\alpha + \beta = 1)$ to tradeoff magnitude difference and time. The solution of (15) returns the best decreasing edge for each increasing edge C_{Pi}.

After pairing all edges between the matched positive and negative cluster, the rejected (unpaired) edges will be included into the next cluster, i.e., the next magnitude-wise smaller cluster and the above iteration carried out again (see Fig. 1).

E. DISAGGREGATION OUTPUT

The above feature matching method forms a pair of matched positive and negative clusters, where each such pair corresponds to one potential appliance state. The matched samples from the two paired clusters define the start and the end of the appliance running event. Each disaggregated event is finally labelled by comparing the disaggregated signature with an existing database of appliance signatures available for that particular household, which can at first be done via crowd-sourcing [42] or a short-time diary, i.e., the signature is extracted at the time-stamp the householder switches on and off the appliances in their house, and after that automatically, for example, as in [23]. If the appliance is not present in the database (there is no match between database signatures and the extracted event using the above feature matching approach), it will be added to the database. If the database is empty, the appliances would be added as they are disaggregated and arbitrary labelled, until the consumer confirms their label.

As is common practice in NALM [19], [20], [23], each appliance load is estimated using the average appliance power consumption during the identified event time interval, although, some more sophisticated approaches are possible [21].

V. PERFORMANCE LIMITS

It is expected that the proposed algorithm will work well if the average load of each appliance is distinct enough from other appliance loads in the house, and if P_m, for each m, does not fluctuate much. Next, we estimate how the mean and variance of the appliance power load affect the disaggregation result. That is, given statistics of the appliance load, we would like to predict if the algorithm will be successful in disaggregating these appliances. Since the proposed approach disaggregates one appliance at a time, to estimate the limits of our approach, without loss of generality, we consider a mixture of two appliance loads.

We model each appliance power load as a two-state Markov source, where in the on-state Source m (corresponding to Appliance m power load) has Gaussian distribution with mean ω_m and variances σ_m^2. In the off-state, the source load is zero. We note that it is widely accepted that for the majority of appliances, the power load follows a Gaussian distribution [12], [27].

That is, for $m = 1, 2$, the i-th sample of Source m is defined as:

$$x_{mi} = \omega_m + n_{mi}, \tag{16}$$

where n_{mi} denotes zero-mean Gaussian "noise" with standard deviation σ_m that models the fluctuation of the source load from its mean. Then, we generate a mixture of the two time-series signals as:

$$y_i = \mu_i x_{1i} + \nu_i x_{2i}, \tag{17}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are binary-valued random vectors, which drive transitions between the on and off state. We initially set both μ_1 and ν_1 to 0, assuming that at the start both appliances are off. The remaining values are generated according to the previous state and the following transition probabilities: $\mathbf{P}(\mu_{i+1} = 1|\mu_i = 1) = 0.95$, $\mathbf{P}(\mu_{i+1} = 1|\mu_i = 0) = 0.03$, $\mathbf{P}(\nu_{i+1} = 1|\nu_i = 1) = 0.94$, $\mathbf{P}(\nu_{i+1} = 1|\nu_i = 0) = 0.05$, which are selected to mimic appliance usage. Since most domestic appliances' operating power is below 3000 Watt (W), both ω_1 and ω_2 are set in this range.

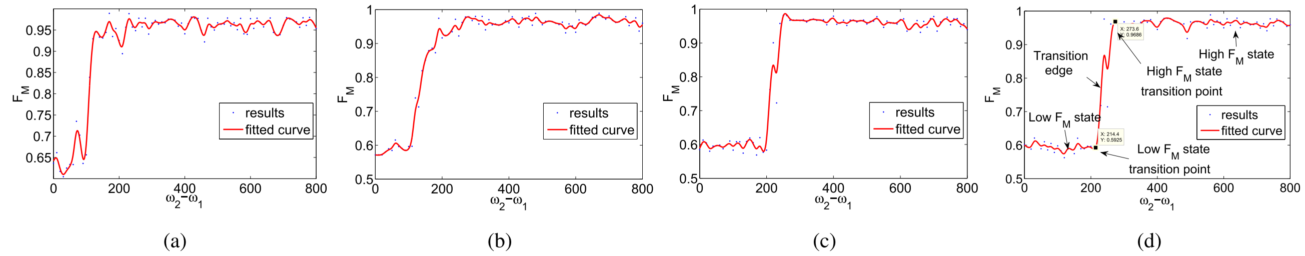
We analyze the performance of the algorithm by looking at five cases. First, we vary $\Delta\omega = \omega_2 - \omega_1$ while keeping the variances fixed. Then, we fixed the means and vary both σ_1 and σ_2. In Case 3, we fix ω_1 and σ_1 and vary ω_2 and σ_2. Then, we vary all four parameters and look at the influence of the scaling factor in the Gaussian Kernel function scaling factor, ρ. We always use the F-Measure, F_M, as performance measure (see the next section for the definition of F_M (23)).

A. CASE 1

Firstly, we fix ω_1, σ_1 and σ_2 and vary ω_2, to see how $\Delta\omega = \omega_2 - \omega_1$ influences the disaggregation results.

In Fig. 2, we present the results for four different values of ω_1, 200, 600, 1000 and 1400W. These values represent a range of typical domestic appliances, from low-power appliances such as fridges and freezers that often operate around 200W, mid-power appliances, such as stoves and the dry mode of air conditioners that usually operate at around 600W, to high-power appliances such as toasters, washing mode of dishwashers (around 1000W) and typical electric kettles and microwaves (1400W).

It can be noticed from each subplot in Fig. 2 that the performance curve of F_M values always has two clearly distinguishable states: a low F_M-value state, for small $\Delta\omega$, when the performance is poor, and a high F_M-value state for large $\Delta\omega$, labeled in Fig. 2(d). Thus, as expected, for the fixed source variances, our disaggregation approach always works well when $\Delta\omega$ is large enough. A sharp increasing edge provides a clear performance limit, when the algorithm accuracy is close to 100% or $F_M = 1$. This result is expected, since the larger the difference in appliance loads making up the aggregate, the better the disaggregation.



(a) (b) (c) (d)

FIGURE 2. F_M **versus** $\Delta\omega$ **for different** ω_1**. Both** σ_1 **and** σ_2 **are fixed to 20. (a)** $\omega_1 = 200$**. (b)** $\omega_1 = 600$**. (c)** $\omega_1 = 1000$**. (d)** $\omega_1 = 1400$**.**

B. CASE 2

Next, ω_1 and ω_2 are set to 600W and 800W, respectively, thus, $\Delta\omega$ is fixed as 200W. We keep changing both σ_1 and σ_2 to find how the variance term affects the performance. Noise variance is equivalent to power load fluctuation of an appliance, that is, the power deviation from the mean value.

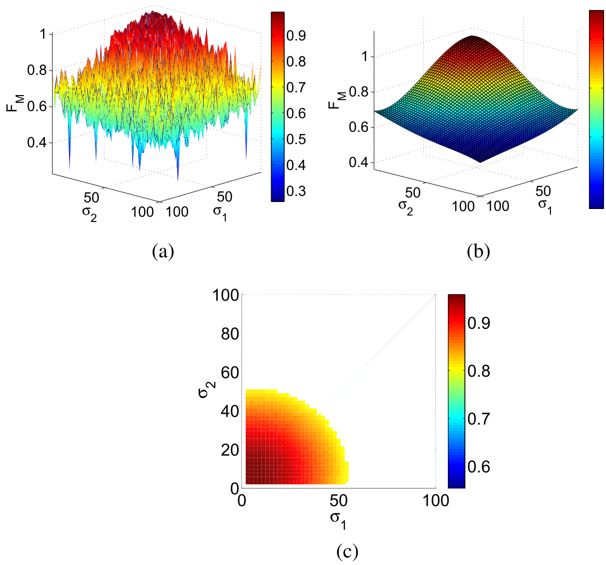


FIGURE 3. F_M **versus** σ**'s where** $\Delta\omega = 200$**. (a)** F_M **results. (b) Fitted surface. (c) Fitted surface selection.**

Fig. 3 presents the F_M performance versus σ_1 and σ_2 and the corresponding fitted surface which is a quartic polynomial consisting of 14 coefficients. Assuming that $F_M > 0.8$ provides acceptable performance, Fig. 3(c) shows the $[\sigma_1, \sigma_2]$ plane where F_M is greater than 0.8, which can be fitted as a quadrant defined by:

$$\sigma_1^2 + \sigma_2^2 \leq r^2, \tag{18}$$

where $\sigma_1 \geq 0$, $\sigma_2 \geq 0$ and $r \approx 52$ in this example. For different ω_1 and ω_2, heuristically, we found that the radius of the fitted quadrant will change, but (18) will still hold. Thus, there is a clear limit in the intensity of variation that the proposed algorithm can tolerate, and this depends on the mean values of power loads.

C. CASE 3

In this step, we fix ω_1 to 600W, σ_1 to 20 and we vary ω_2 and σ_2 and investigate how the performance is affected with respect to $\Delta\omega$ and $\frac{\sigma_2}{\sigma_1}$. Since we know from Fig. 2 that there exists a value of $\Delta\omega$ when F_M sharply increases, we search for the two planes obtained by varying the ratio $\frac{\sigma_2}{\sigma_1}$.

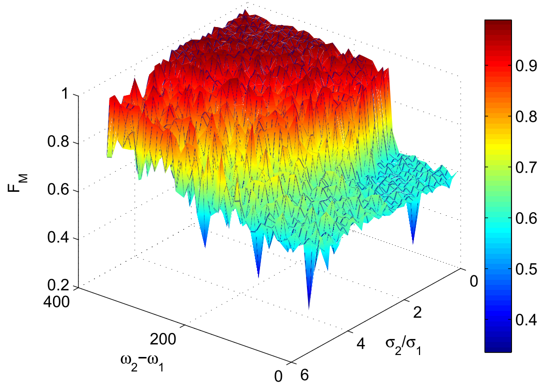


FIGURE 4. F_M **versus** $\Delta\omega = \omega_2 - \omega_1$ **and** $\frac{\sigma_2}{\sigma_1}$**.**

Both high F_M "plane" and low F_M "plane" are clearly shown in Fig. 4. When $\frac{\sigma_2}{\sigma_1}$ is less than 2, the separation edge between high and low F_M "plains" is steep. It is more moderate when $\frac{\sigma_2}{\sigma_1}$ becomes larger, and the edge almost disappears when $\frac{\sigma_2}{\sigma_1}$ is around 5. So, if the variance ratio is too high, the algorithm will not perform well, which is expected since one ("noisier") source will significantly affect disaggregation of the other.

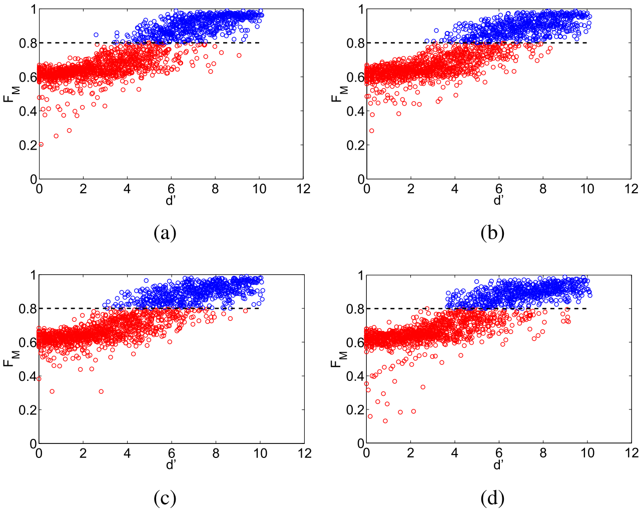
D. CASE 4

Next we investigate the performance of the proposed algorithm as a function of the scaling factor, ρ, in the Gaussian kernel weighting function. To capture the changes in all four parameters, we use the sensitivity index d' defined by:

$$d' = \frac{|\omega_2 - \omega_1|}{\sqrt{\frac{1}{2}(\sigma_1^2 + \sigma_2^2)}}. \tag{19}$$

We can see the F_M performance versus d' in Fig. 5 for four different initial ρ in the Gaussian kernel weighting function.

Fig. 5 clearly shows that, as expected, the greater the d', the better the performance. In the figure, the blue points



(a) (b)

(c) (d)

FIGURE 5. F_M **versus** $\Delta\omega$ **for different** ω_1**. (a)** $\rho = 20$**. (b)** $\rho = 25$**. (c)** $\rho = 30$**. (d)** $\rho = 35$**.**

represent the cases of $F_M \geq 0.8$. Note that our approach always performs poorly when d' is less than 3.

It seems that ρ does not affect the results. In fact, the iterative nature of our clustering approach reduces the effect of ρ. One can see from Table 1, that by increasing ρ, the number of clusters before and after pairing change, but the F_M values stay similar.

TABLE 1. **Performance of the proposed approach and the number of clusters for different initial** ρ**.**

ρ	$\mathrm{F_M}$	No. clust. before pairing	No. clust. after pairing
5	0.98	17	6
25	0.98	8	6
45	0.98	6	4
65	0.97	7	6
85	0.97	7	6
105	0.98	5	4

E. CASE 5

Finally, we focus on the distribution of high F_M transition state points, when F_M becomes acceptably high, namely, the limit of the performance of our approach. We fixed ω_1 at 1000W, and we keep changing σ_1 and σ_2 in the range from 2 to 30, with steps of 2. Fig. 6 shows the high F_M state transition points, that is, the minimum $\Delta\omega$ required for good disaggregation performance, denoted by $\Delta\omega_0$.

In Fig. 6(a), the value of the bound $\Delta\omega_0$ generally increases when either σ_1 or σ_2 increases. Since the definition of sensitivity index in (19) indicates ω's and σ's have the same dimension, we fit the boundary as a three-variable linear equation which represent a plane in Fig. 6(b) as:

$$\Delta\omega_0 = a\sigma_1 + b\sigma_2 + c \tag{20}$$

where a is 1.67, b is 2.46 and c is 188. This equation will be used in the next section as a performance limit to predict disaggregation performance of the proposed algorithm.

F. SUMMARY

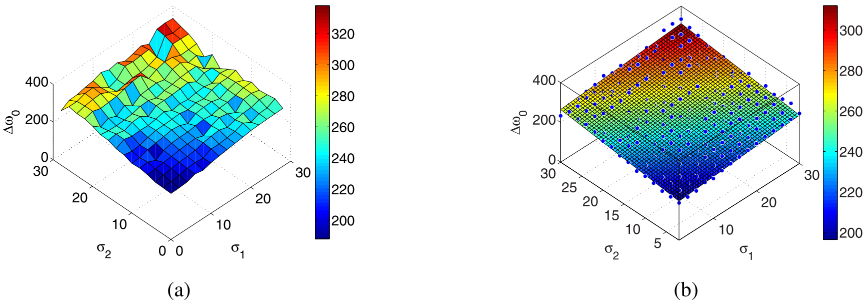
From the above analysis we can make following conclusions:

- there exists a clear $\Delta\omega$ limit between good algorithm performance and poor performance;
- the fluctuation of an appliance power load influences algorithm performance;
- when the ratio between the variances of the appliance load is high, the performance is poor;
- the performance limits can be expressed adequately using sensitivity measure d';
- for given variances of appliance loads, the minimum $\Delta\omega$ required for good performance can be estimated using (20).

Consequently, with the above derived limits of the proposed algorithm, we can predict its performance as well as explain both the success and failure of the approach given appliance load statistics.

VI. CASE STUDY

In this section we clarify how the proposed algorithm works in practice, from the generation of the graph signal to final feature matching. As input, we use the aggregate active power readings of House 2 from REDD dataset [17] collected from April 28th 2011 to April 30th 2011, downsampled to 1 minute, as a typical example that does not have any gaps. The aggregate power readings are shown in Fig. 11 (red curve).



(a) (b)

FIGURE 6. **The limits of the proposed algorithm. All values on and above the surface lead to** $F_M \geq 0.8$**. (a) Actual results. (b) Fitted surface.**

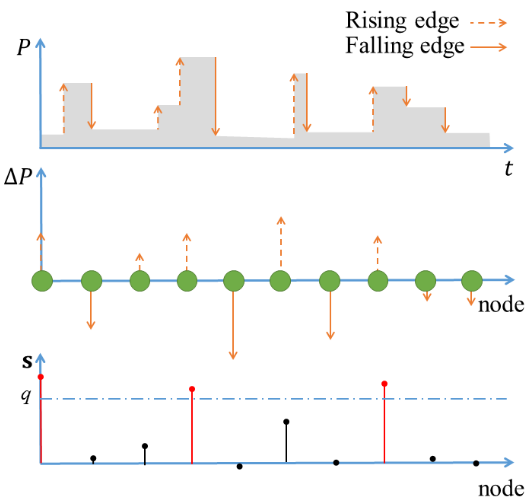


FIGURE 7. Graph generation for Stage 1 of clustering.

In order to minimize the effects of load fluctuations without losing the state transition edges of all appliances, we first extract the events (candidate edges) by thresholding (10) on the aggregate load with $T_0 = 10$W. A segment of the resulting events is shown in Fig. 7 (top) as aggregate power vs. time.

A. GRAPH GENERATION

All rising and falling edges are indexed by the nodes of a graph (see the middle figure in Fig. 7). Each power edge ΔP_i in Fig. 7 (top) corresponds to one node v_i in the graph and the edge between nodes v_i and v_j is assigned using (1) with $x_i = \Delta P_i$ and $x_j = \Delta P_j$ and Euclidean distance measure.

B. INITIAL CLUSTERING

The task is to group all edges similar to the first edge into the same cluster. To do that, a graph signal $\mathbf{s}$ is formed where each sample of $\mathbf{s}$ corresponds to one node in the graph. $s(1)$ is set to 1 and $\mathbf{s}^*_{2:n}$ are calculated by (4) to ensure that the graph signal remains smooth. This step is intuitive, since we expect smooth changes between the edges within the same cluster. Finally, we group all candidate edges which are magnitude-wise similar to the first edge by thresholding of $\mathbf{s}^*_{2:n}$ using a high threshold $q = 0.98$ (i.e., Edge i, $i > 1$, will be grouped in the cluster represented by the first edge only if $\frac{s_i}{s_1} \geq 0.98$), displayed as red edges in Fig. 7 (bottom). Note that time duration information is not a feature in this step. We repeat the same clustering procedure for the unclustered events until all events are clustered. As a result, in this example, eight ‘‘initial’’ clusters are formed shown in Fig. 8 (left).

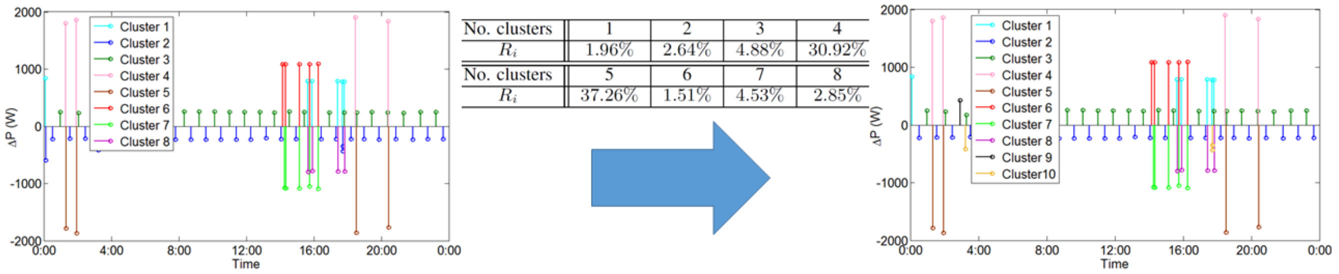
C. REFINED CLUSTERING AND ADAPTIVE THRESHOLDING

For each cluster, RSD R_i is used to denote the quality of the cluster defined in (11). The clusters with $R_i > K$, where $K = 10\%$, will be sub-clustered to make sure that each cluster has a high precision degree and is likely to contain the events of only one power state (Fig. 8 (middle)). For example, a cluster with mean magnitude value of 150W should have a standard deviation of at most 15W to be an acceptable cluster. In this example, Clusters No.4 and No.5 need sub-clustering by halving ρ in Gaussian kernel weighting function for enhancing the clustering degree. At the end of clustering, small-sized clusters are merged into larger magnitude-wise clusters to keep a balance between the number of positive and negative clusters. After sub-clustering and merging, 10 clusters are obtained (see Fig. 8 (right)).

D. FEATURE MATCHING

In the next step, we match each positive cluster to the magnitude-wise closest negative cluster. Then for each positive-negative cluster pair, we pair each rising edge to an optimal falling edge using feature matching, as explained in Subsection IV-D. This is demonstrated in Fig. 9, where red and blue impulses show positive and negative edges, respectively, and are refrigerator state transitions. The task is to find a falling edge corresponding to each rising edge. In this example, for one particular rising edge, two possible falling edge candidates are identified based on their time distance to the rising edge, identified within the dashed-line rectangles.

Fig. 10 demonstrates how pairing is performed using GSP, where both power and time information are treated as features. There exist two candidate falling edges at distances Δt_1 and Δt_2, from the rising edge (see Fig. 10 (top)). We build two graphs: (1) one to capture the similarity between absolute power values of the rising edge and candidate falling edges (the left-hand side on the second figure from the top in Fig. 10); (2) the second graph captures time duration



No. clusters	1	2	3	4
R_i	1.96%	2.64%	4.88%	30.92%
No. clusters	5	6	7	8
R_i	37.26%	1.51%	4.53%	2.85%

FIGURE 8. Clustering results.

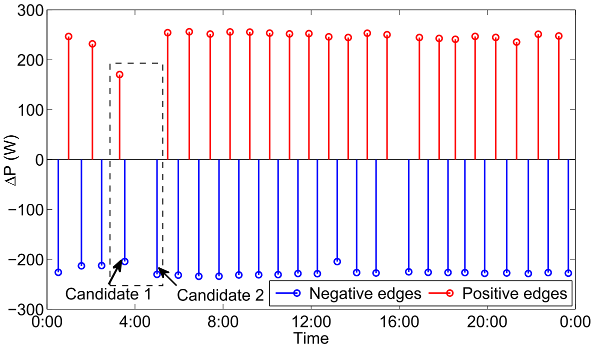


FIGURE 9. A typical positive-negative cluster pair.

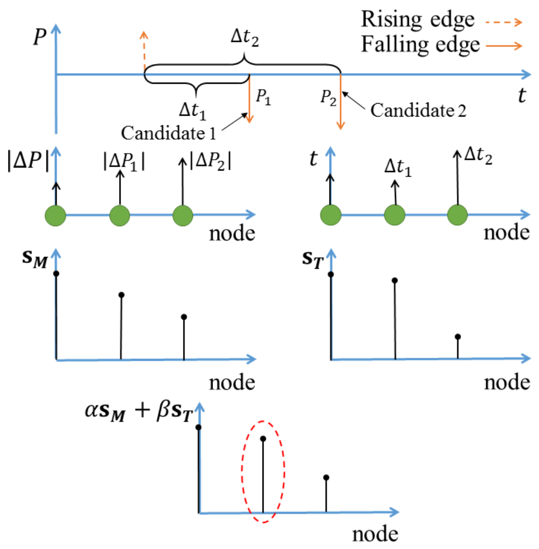


FIGURE 10. Graph example for feature matching.

between the rising edge and the candidate falling edges (the right-hand side on the second figure from the top in Fig. 10).

In the absolute power value graph, the first node's value is set to the difference between the mean of absolute values

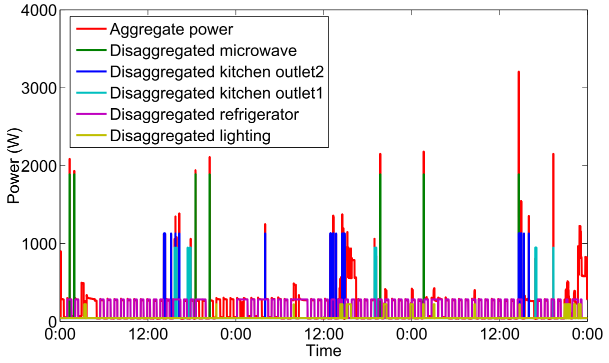
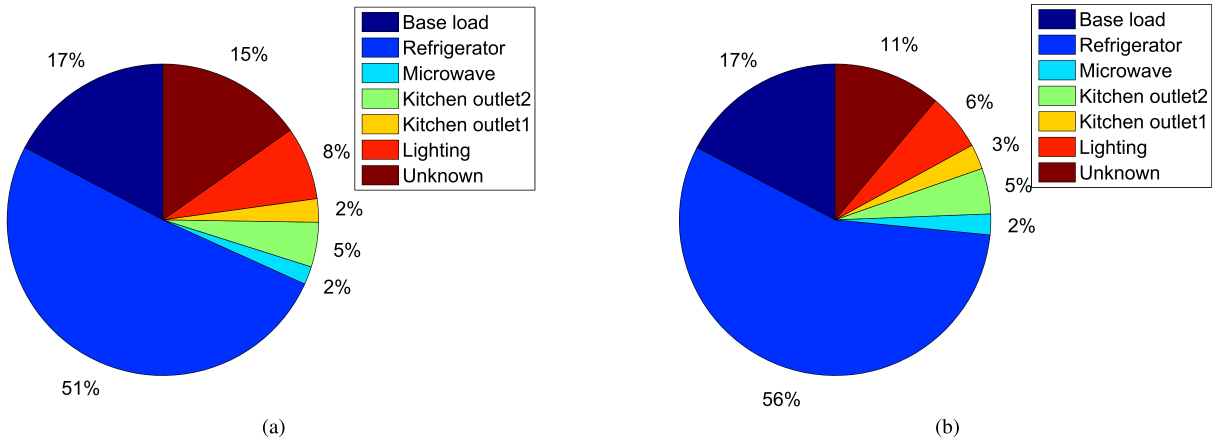


FIGURE 11. Disaggregation result for three typical days.

of all rising edges and the mean of the absolute values of all falling edges. The remaining nodes index the differences in absolute power value between the rising edge and the candidate falling edges. In the time interval graph, the first node is set as the median of the set containing the time interval between each rising edge and the following falling edge. The remaining nodes index the time duration between the rising edge and the falling edge candidates. The graph global smoothness minimization solutions $\mathbf{s}_{\mathbf{M}}^{*}$ and $\mathbf{s}_{\mathbf{T}}^{*}$ for both power and time interval information are calculated using (7) and adjacency matrices $\mathbf{A}_{\mathbf{M}}$ and $\mathbf{A}_{\mathbf{T}}$ are obtained through (13) and (14), respectively. See the third row in Fig. 10, where the left-hand side shows power graph signal and the right-hand side the time interval graph signal. In the last step, we weight and then sum the smoothness minimization solution of both graphs using (15), where the maximum value corresponds to the optimal event pair. The optimal falling edge corresponding to the rising edge is circled in Fig. 10 (bottom).

The disaggregation result for the three typical days illustrated by this example is shown in Fig. 11. Note that there is a good alignment between the true aggregate values and the disaggregated result. Fig. 12 presents the NALM accuracy for the three typical days. base load is defined as



(a)

(b)

FIGURE 12. Pie charts of NALM results for House 2 from REDD dataset for three typical days. (a) Disaggregation results. (b) Ground truth.

the minimum power value over the three monitored days. The disaggregation error between the estimated (12.59kWh) and the real energy consumption (13.2kWh) is 4.6%. Unknown in the left pie chart denotes the difference between the true aggregate consumption and the sum of the disaggregated load, and indicates the load that cannot be disaggregated. Unknown percentage in the right, ground truth, pie chart is the difference between the measured aggregate consumption and the sum of loads of the individual, sub-metered appliances.

VII. RESULTS AND DISCUSSION

We demonstrate the performance of the proposed algorithm using active power readings from two open-access datasets: (1) the REDD dataset [17] of US houses, downsampled to 1 minute; (2) the UK REFIT Electrical Loads dataset [18], with measurements collected every 8 seconds. Both datasets are supported by NILMTK [44] - a toolkit designed to help researchers evaluate the accuracy of NALM algorithms.

The REDD dataset is widely used for the evaluation of various NALM approaches [10], [14], [16], [23], [31], [38]. REDD houses contain several appliances with a small unknown load. The REFIT dataset, on the other hand, is more challenging, as the houses contain numerous unknown appliances. The results are shown only for appliances for which time-diaries are collected or whose usage was measured at appliance level via appliance monitors; these appliances are referred to as known appliances.

We empirically set K for evaluating cluster quality, i.e., RSD R_i, to 10% to minimize $|R_i - K|$ for all appliances. Initial threshold T_0 is set to 10W to avoid detecting stand-by states without filtering any appliances' state-transition edge. The initial Gaussian kernel weighting function scaling factor ρ is set to 20 to avoid over-clustering for most domestic appliances, which work below 3000W. $\alpha = \beta = 0.5$ as power and time have the same influence when pairing rising and falling edges, and q is set to 0.98 to group only highly correlated samples, and reduce falsely clustered edges.

A. PERFORMANCE MEASURE

The evaluation metrics used are adapted from [10], [27], *Precision* (PR), *Recall* (RE) and *F-Measure* (F_M). As in [27], we separate true positives (TP) into two cases, accurate true positive (ATP) and inaccurate true positive (ITP). ATP presents the correct claim the detected appliance was running and the corresponding events are correctly named, while, ITP represents the correct claim the detected appliance was running but the corresponding events are wrongly named. False positives (FP) denote a wrong claim that the detected appliance was running, and false negatives (FN) indicate that the appliance operation was not detected. Then:

$$PR = ATP/(ATP + FP) \tag{21}$$

$$RE = ATP/(ATP + ITP + FN) \tag{22}$$

$$F_M = 2 \cdot (PR \cdot RE)/(PR + RE), \tag{23}$$

PR represents the accuracy of event detection, thus a lower FP leads to a higher PR. RE denotes the strength of events detection and clustering, thus lower FN and ITP lead to a higher RE. F_M balances PR and RE.

For comparison with recent approaches [14], [31], we also use $Acc.$ to demonstrate disaggregation accuracy, defined as,

$$Acc. = 1 - \frac{\sum_{i=1}^{n} \sum_{m \in \mathcal{M}} |\hat{P}_{m_{t_i}} - P_{m t_i}|}{2 \sum_{i=1}^{n} \bar{P}_{t_i}}, \tag{24}$$

where n is the number of test samples, $\hat{P}_{m_{t_i}}$ refers to the estimated power of appliance m at time instance t_i, P_{mt_i} is the actual power consumption by appliance m at time instance t_i, and $\bar{P}_{t_i}$ denotes the observed total power consumption at time instance t_i.

B. ALGORITHM PERFORMANCE

In order to enable like-for-like comparison with [10], [12], [14], [23], and [38], we use REDD Houses 1, 2 and 6. House 2 contains the fewest power states among all three houses and has a low appliance complexity and low time complexity as defined in [43], implying that House 2 is relatively the easiest house to disaggregate. House 1 has a similar number of appliances as House 2, but more power states [43], and consequently is more challenging to disaggregate. House 6 has the largest number of appliances of the three. Approximately, two weeks worth of consecutive data is used in this experiment. Note that all data is used for testing - there is no training in the proposed method.

TABLE 2. Performance of the proposed approach for House 1 from the REDD dataset.

Appliance	ATP	ITP	FP	FN	PR	RE	F_M
Microwave	174	105	21	13	0.89	0.6	0.72
Kitchen outlet	19	14	1	13	0.95	0.41	0.57
Washer dryer	191	31	11	78	0.95	0.64	0.76
Oven	29	4	2	26	0.94	0.49	0.64
Lighting	35	26	16	32	0.69	0.38	0.49
Refrigerator	513	50	20	127	0.96	0.74	0.84
Dishwasher	49	139	3	15	0.94	0.24	0.38
Bathroom GFI	39	11	9	10	0.81	0.65	0.72

Table 2 shows the performance for REDD House 1. Relatively low F_M results for dishwasher can be attributed to similarity between the refrigerator load and one power state of the dishwasher. Indeed, the average working power of refrigerator is 201W and that of the low power state of the dishwasher is 210.6W, hence the two power states are likely to be grouped into the same cluster, as shown by the large ITP, which captures detected events that are wrongly labelled as refrigerator.

The results for House 2 and 6 are shown in Tables 3 and 4, respectively. From Table 3, we can see that most appliances are disaggregated with accuracy >70%. The worst performance is achieved for the stove, due to a high FP. The poor performance for the heater and lighting in House 6, shown in Table 4, is due to the questionable ground truth

TABLE 3. Performance of the proposed approach for House 2 from the REDD dataset.

Appliance	ATP	ITP	FP	FN	PR	RE	F_M
Microwave	80	5	0	5	1	0.89	0.94
Kitchen outlet1	39	0	1	13	0.98	0.75	0.85
Kitchen outlet2	172	9	11	18	0.94	0.86	0.85
Stove	34	0	83	4	0.29	0.89	0.44
Refrigerator	595	17	110	161	0.84	0.77	0.8
Dishwasher	19	15	11	13	0.63	0.4	0.49
Lighting	70	0	110	9	0.64	0.89	0.74

TABLE 4. Performance of the proposed approach for House 6 from the REDD dataset.

Appliance	ATP	ITP	FP	FN	PR	RE	F_M
Microwave	10	0	3	0	0.77	1	0.87
Kitchen outlets	4	1	3	3	0.57	0.5	0.53
Stove	7	5	3	2	0.7	0.5	0.58
Refrigerator	439	8	56	132	0.89	0.76	0.82
Electronics	26	6	61	5	0.3	0.7	0.42
Heater	3	0	56	3	0.05	0.5	0.09
Air Conditioner	44	9	0	1	1	0.81	0.9
Lighting	7	6	7	12	0.5	0.28	0.36
Outlets unknown	146	6	56	65	0.72	0.67	0.69

data collected using sub-metering for these two appliances (sub-metering data is very noisy with many spikes).

TABLE 5. Performance of the proposed approach for House 8 from the REFIT dataset.

Appliance	ATP	ITP	FP	FN	PR	RE	F_M
Microwave	7	10	0	3	1	0.35	0.52
Toaster	4	1	2	1	0.67	0.67	0.67
Kettle	39	7	6	2	0.87	0.81	0.84
Refrigerator	18	0	2	0	0.9	1	0.95
Freezer	54	16	180	24	0.23	0.57	0.32
TV	4	0	180	6	0.02	0.4	0.04
Washing Machine	3	1	8	0	0.27	0.75	0.4

Next, the results for REFIT House 8 are presented in Table 5, disaggregated over a period of three consecutive days when all seven known appliances were running. It can be seen that the proposed training-less approach showed very good performance for the kettle and refrigerator, but poor results for the TV, caused by TV being grouped in the same cluster as the freezer, because TV and freezer have close active power range. Note that, our approach shows average performance (across all appliances) of $F_M = 0.49$, which is better than the unsupervised HMM-based approach in [23], which uses training and shows $F_M = 0.46$ for the same house.

The algorithms were implemented in Matlab2014 and were executed on Intel Core i7-4700MQ CPU 2.40GHz machine running Windows 8.1. It takes 65.6 seconds to complete data filtering and clustering and 3.5 seconds to finish feature matching, i.e., 69.1 seconds in total, for processing 20,160 samples, which means roughly 3 milliseconds per sample. Note that optimized implementations would require less time.

C. COMPARISON WITH STATE OF THE ART

In this section, we compare our results with some state-of-the-art NALM algorithms, proposed for low sampling rates and active power measurements.

First, we compare our results, F_{M_U}, to those of the supervised GSP-based approach in [10], F_{M_S}, and the results using the unsupervised HMM-based method of [12], F_{M_H}, as reported in [10]. All three methods are tested using the same data, while the two benchmark methods require additional data for training [10], [12].

TABLE 6. Performance of three NALM approaches for Houses 2 and 6 from the REDD dataset.

Appliance	House 2			House 6		
	F_{M_U}	F_{M_S}	F_{M_H}	F_{M_U}	F_{M_S}	F_{M_H}
Microwave	0.94	0.26	0.47	0.87	0.92	0
Kitchen outlet2	0.85	0.59	0.68	0.53	1	0
Stove	0.44	0.41	0.21	0.58	1	0
Refrigerator	0.8	0.63	0.9	0.82	0.54	0.88
Dishwasher	0.49	0.56	0.04	0.42	-	-
Heater	-	-	-	0.09	0.11	0.03
Air Conditioner	-	-	-	0.9	0.49	0.12

The results for the three NALM approaches are shown in Tables 6 for the REDD datasets. Overall, both GSP-based approaches perform significantly better than the HMM-based approach. The proposed unsupervised GSP-based approach performs, on average, as well as the supervised one of [10], but without the training and supervised labelling overhead.

TABLE 7. Performance comparison with [14] and [31] for House 2 from the REDD dataset. Note that only 5 top-consuming appliances are disaggregated in [14], while the proposed methods disaggregated seven appliances.

Approach	*Acc.*
Proposed method	77.2%
EM FHMM [14]	50.8%
F-HDP-HMM [14]	70.7%
F-HDP-HSMM [14]	84.8%
FHMM (without interaction) [31]	65.8%
FHMM (with interaction) [31]	66.5%

The performance of the proposed method for the REDD houses is comparable to the three FHMM-based unsupervised approaches of [14] and two FMM-based approaches of [31] that all require training. Indeed, from Table 7 that shows the accuracy *Acc.* for House 2 from REDD dataset, we can observe that the proposed method is more accurate than Expectation Maximization FHMM (EM-FHMM), Factorial Hierarchical Dirichlet Process (F-HDP) HMM as well as both FHMM approaches with and without interaction and performs slightly worse than F-HDP Hidden Semi-Markov Model (F-HDP-HSMM) approach of [14]. Note, however, that in [14], only top five power-drawing appliances are disaggregated, while both our algorithm and that in [31] disaggregate seven appliances in House 2.

Next, we demonstrate the performance of our approach for House 1 from REDD dataset with ground truth shown

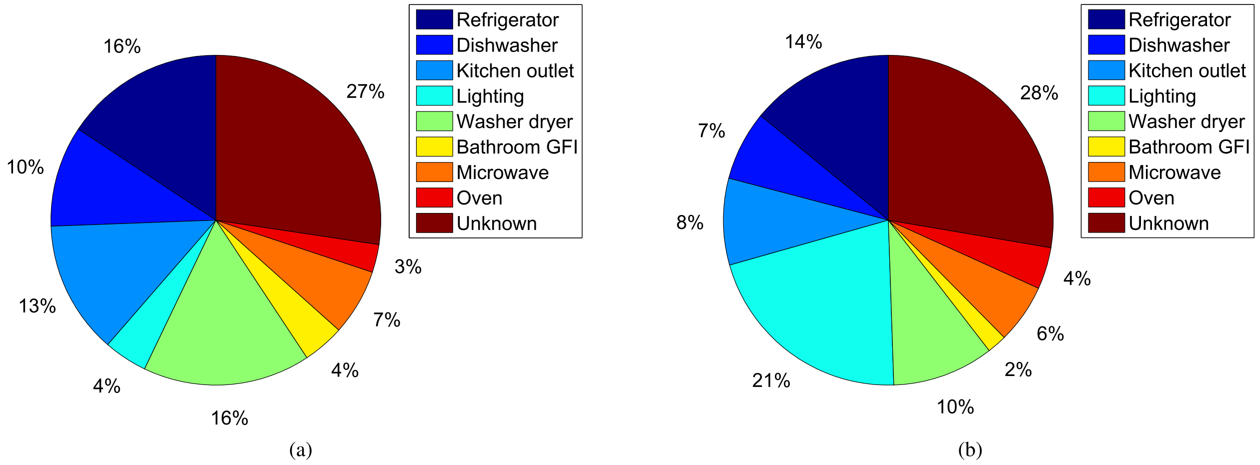


FIGURE 13. **Pie charts of NALM results for House 1 from REDD datasets. (a) Disaggregation results. (b) Ground truth.**

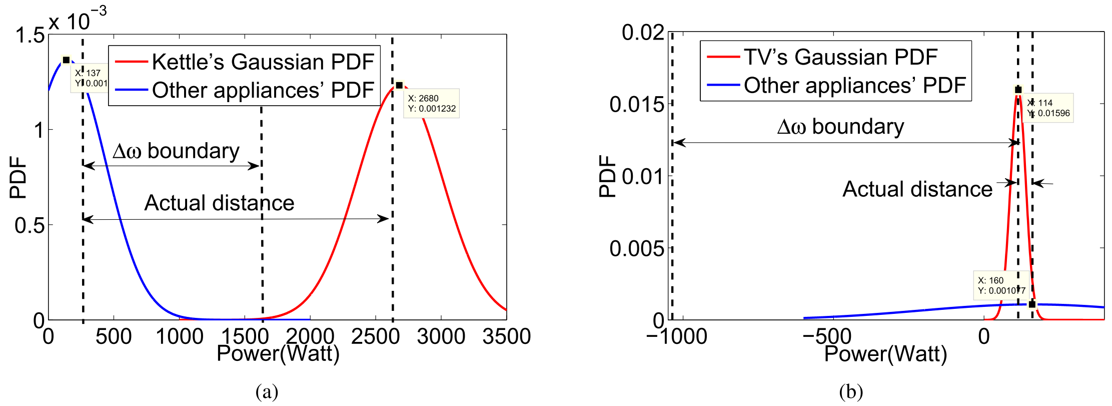


FIGURE 14. **REFIT House 8 kettle and TV disaggregation. (a) Kettle disaggregation. (b) TV disaggregation.**

in Fig. 13(b) for comparison with a recent unsupervised approach in [38]. In contrast to [38], which uses the aggregated load of six measured individual appliances, we apply the proposed algorithm to the measured aggregate data from the REDD dataset, downsampled to 1 minute, to demonstrate robustness to noise. Disaggregation is done for a period of 17 consecutive days. The resulting estimation error between the estimated (58.32kWh) and the real energy consumption (60.16kWh) is 3%, in the presence of noise, as oppesed to the reported 2% in [38]. Additionally, we identify eight appliances in contrast to three 'virtual' appliances (i.e., appliances with similar power demand are grouped together) in [38].

D. PERFORMANCE BOUND ANALYSIS

Next, we use Equation (20) to predict the performance of the proposed algorithm. We demonstrate the usefulness of the analytical study and the bound (20) for two appliances from the REFIT dataset: kettle, where disaggregation was successful, and TV where the algorithm did not work (see Table 5). First, we model the power load of each appliance using the Gaussian probability density function (PDF) (see Fig. 14), as is common practice [12], [27].

The kettle's standard deviation is 323.8, while standard deviation of the total aggregate signal without kettle is 292.2 and the actual difference in means of the two Gaussian distributions is found to be $\Delta\omega$ is 2537.1W which is greater than the $\Delta\omega$ boundary as shown in Fig. 14 (a). According to (20) the performance limit for $\Delta\omega_0$ is much lower - 1472.5W indicating correctly that disaggregation will work. From Table 5, the F_M value of kettle disaggregation is 0.84>0.8, which confirms our estimate.

Fig. 14 (b) represents TV disaggregation. As standard deviation of TV and the remaining load, σ_1 and σ_2, are equal to 25 and 370.5, respectively, the $\Delta\omega_0$ limit given by (20) is 1141.2W. As, the actual distance $\Delta\omega$ in this case is 46.8W, which is much smaller than the limit, we predict that the performance will be bad. Since the F_M value of TV in House 8 in Table 5 is 0.04, the poor performance confirms our prediction.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we build on the emerging GSP concepts to develop a novel, *blind*, unsupervised low-rate NALM approach. The main motivation comes from the fact that GSP does not require training, can accurately capture signal patterns that occur rarely, is robust to noisy data, and has low implementation complexity.

Based on the results from disaggregating aggregate loads measured from 4 real houses, we show that our training-less GSP-based NALM approach has comparable performance with the supervised GSP-based NALM approach of [10] outperforming the unsupervised HMM-based method. We heuristically determined the performance limits of the proposed algorithm and demonstrated the usefulness of this limit to estimate the disaggregation performance.

Due to its low complexity, simple operation and minimal customer input (for initial labelling), our algorithm can be applied on large scale as an embedded system as part of Consumer Access Device [5] with an online feedback interface that users can access.

This paper has further demonstrated the potential of GSP for load disaggregation. Future work will include improving event detection via pre-processing (denoising and filtering), enhancing the robustness of the algorithm when dealing with appliance simultaneous operation and improving the performance for multi-state appliances by incorporating time of use as a feature before clustering and inter-connection between power state occurrence. Furthermore, in the current algorithm, if a single state transition lasts longer than the sampling period multiple consecutive events will be identified. Future work will be focused on edge detection where one state change lasts longer than the sampling duration.

REFERENCES

[1] B. Zhao, L. Stankovic, and V. Stankovic, "Blind non-intrusive appliance load monitoring using graph-based signal processing," in *Proc. 3rd IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Orlando, FL, USA, Dec. 2015, pp. 1–5.

[2] G. W. Hart, *Nonintrusive Appliance Load Data Acquisition Method*. Cambridge, MA, USA: MIT Energy Laboratory, Sep. 1984.

[3] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert, "Is disaggregation the holy grail of energy efficiency? The case of electricity," *Energy Policy*, vol. 52, pp. 213–234, Jan. 2013.

[4] L. Stankovic, C. Wilsaon, J. Liao, V. Stankovic, R. Hauxwell-Baldwin, and M. Coleman, "Understanding domestic appliance use through their linkages to common activities," in *Proc. 8th Int. Conf. Energy Efficiency Domestic Appl. Lighting (EEDAL)*, Horw, Switzerland, Aug. 2015, pp. 1–14.

[5] "Smart metering implementation programme," Dept. Energy Climate Change, London, U.K., Tech. Rep. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/68898/smart_meters_equipment_technical_spec_version_2.pdf

[6] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[7] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[8] A. Sandryhaila and J. Moura, "Classification via regularization on graphs," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Austin, TX, USA, Dec. 2013, pp. 495–498.

[9] C. Yang, G. Cheung, and V. Stankovic, "Estimating heart rate via depth video motion tracking," in *Proc. Int. Conf. Multimedia Expo (ICME)*, Turin, Italy, Jun./Jul. 2015, pp. 1–6.

[10] V. Stankovic, J. Liao, and L. Stankovic, "A graph-based signal processing approach for low-rate energy disaggregation," in *Proc. IEEE Symp. Comput. Intell. (SSCI)*, Orlando, FL, USA, Dec. 2014, pp. 81–87.

[11] C. Zhang, D. Florêncio, and P. A. Chou, "Graph signal processing—A probabilistic framework," Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2015-31, 2015. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=243326

[12] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "Non-intrusive load monitoring using prior models of general appliance types," in *Proc. 26th Conf. Artif. Intell. (AAAI)*, Toronto, ON, Canada, Jul. 2012, pp. 356–362.

[13] J. Z. Kolter and T. Jaakkola, "Approximate inference in additive factorial HMMs with application to energy disaggregation," in *Proc. 15th Int. Conf. Artif. Intell. Statist. (AISTATS)*, La Palma, Canary Islands, Apr. 2012, pp. 1472–1482.

[14] M. J. Johnson and A. S. Willsky, "Bayesian nonparametric hidden semi-Markov models," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 673–701, Feb. 2013.

[15] D. Egarter, V. P. Bhuvana, and W. Elmenreich, "PALDi: Online load disaggregation via particle filtering," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 2, pp. 467–477, Feb. 2015.

[16] O. Parson, S. Ghosh, M. Weal, and A. Rogers, "An unsupervised training method for non-intrusive appliance load monitoring," *Artif. Intell.*, vol. 217, pp. 1–19, Dec. 2014.

[17] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *Proc. Workshop Data Mining Appl. Sustain. (SIGKDD)*, San Diego, CA, USA, 2011, pp. 1–6.

[18] D. Murray *et al.*, "A data management platform for personalised real-time energy feedback," in *Proc. 8th Int. Conf. Energy Efficiency Domestic Appl. Lighting (EEDAL)*, Horw, Switzerland, Aug. 2015, pp. 1–15.

[19] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," *IEEE Trans. Consum. Electron.*, vol. 57, no. 1, pp. 76–84, Feb. 2011.

[20] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," *Sensors*, vol. 12, no. 12, pp. 16838–16866, Dec. 2012.

[21] S. Giri and M. Bergés, "An energy estimation framework for event-based methods in non-intrusive load monitoring," *Energy Convers. Manage.*, vol. 90, pp. 488–498, Jan. 2015.

[22] H. Altrabalsi, J. Liao, L. Stankovic, and V. Stankovic, "A low-complexity energy disaggregation method: Performance and robustness," *AIMS Energy*, vol. 4, no. 1, pp. 884–905, Jan. 2016.

[23] J. Liao, G. Elafoudi, L. Stankovic, and V. Stankovic, "Non-intrusive appliance load monitoring using low-resolution smart meter data," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Venice, Italy, Nov. 2014, pp. 535–540.

[24] J. Z. Kolter, S. Batra, and A. Y. Ng, "Energy disaggregation via discriminative sparse coding," in *Proc. 24th Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, Vancouver, BC, Canada, Dec. 2010, pp. 1153–1161.

[25] N. Pathak, N. Roy, and A. Biswas, "Iterative signal separation assisted energy disaggregation," in *Proc. 6th Int. Green Comput. Conf., Sustain. Comput. Conf. (IGSC)*, Dec. 2015, pp. 1–8.

[26] R. Bonfigli, S. Squartini, M. Fagiani, and F. Piazza, "Unsupervised algorithms for non-intrusive load monitoring: An up-to-date overview," in *Proc. IEEE 15th Int. Conf. Environ. Elect. Eng. (EEEIC)*, Jun. 2015, pp. 1175–1180.

[27] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han, "Unsupervised disaggregation of low frequency power measurements," in *Proc. 11th SIAM Int. Conf. Data Mining*, Mesa, AZ, USA, Apr. 2011, pp. 747–758.

[28] M. Zhong, N. Goddard, and C. Sutton, "Signal aggregate constraints in additive factorial HMMs, with application to energy disaggregation," in *Proc. 28th Annu. Conf. Neural Inform. Process. Syst. (NIPS)*, Montreal, QC, Canada, Dec. 2014, pp. 3590–3598.

[29] S. Pattem, "Unsupervised disaggregation for non-intrusive load monitoring," in *Proc. 11th Int. Conf. Mach. Learn. Appl. (ICMLA)*, vol. 2. Dec. 2012, pp. 515–520.

[30] F. Paradiso, F. Paganelli, D. Giuli, and S. Capobianco, "Context-based energy disaggregation in smart homes," *Future Internet*, vol. 8, no. 1, p. 4, 2016.

[31] M. Aiad and P. H. Lee, "Unsupervised approach for load disaggregation with devices interactions," *Energy Buildings*, vol. 116, pp. 96–103, Mar. 2016.

[32] K. Anderson, J. M. F. Moura, and M. Berges, "Unsupervised approximate power trace decomposition algorithm," in *Proc. 2nd Int. Non-Intrusive Appliance Load Monitor. Workshop*, Austin, TX, USA, Jun. 2014, pp. 5–8.

[33] H. Shao, M. Marwah, and N. Ramakrishnan, "A temporal motif mining approach to unsupervised energy disaggregation: Applications to residential and commercial buildings," in *Proc. 27th AAAI Conf. Artif. Intell.*, Bellevue, WA, USA, Jul. 2013, pp. 1–7.

[34] J. Kelly and W. Knottenbelt, "Neural NILM: Deep neural networks applied to energy disaggregation," in *Proc. 2nd ACM Int. Conf. Embedded Syst. Energy-Efficient Built Environ.*, 2015, pp. 55–64.

[35] N. Batra, A. Singh, and K. Whitehouse. (Oct. 2015). "Neighbourhood NILM: A big-data approach to household energy disaggregation." [Online]. Available: http://arxiv.org/abs/1501.02954

[36] B. Wild, K. S. Barsim, and B. Yang, "A new unsupervised event detector for non-intrusive load monitoring," in *Proc. 3rd IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Orlando, FL, USA, Dec. 2015, pp. 73–77.

[37] H. Gonçalves, A. Ocneanu, and M. Bergés, "Unsupervised disaggregation of appliances using aggregated consumption data," in *Proc. 1st KDD Workshop Data Mining Appl. Sustain. (SustKDD)*, Aug. 2011, pp. 1–6.

[38] D. Egarter and W. Elmenreich, "Autonomous load disaggregation approach based on active power measurements," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, St. Louis, MO, USA, Mar. 2015, pp. 293–298.

[39] S. Barker, S. Kalra, D. Irwin, and P. Shenoy, "NILM redux: The case for emphasizing applications over accuracy," in *Proc. 2nd Int. Non-Intrusive Appliance Load Monitor. Workshop*, Austin, TX, USA, Jun. 2014, pp. 1–4.

[40] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.

[41] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[42] J. Gao, S. Giri, E. C. Kara, and M. Berges, "PLAID: A public dataset of high-resoultion electrical appliance measurements for load identification research: Demo abstract," in *Proc. 1st ACM Conf. Embedded Syst. Energy-Efficient Buildings*, 2014, pp. 198–199.

[43] D. Egarter, M. Pöhacker, and W. Elmenreich. (Jan. 2015). "Complexity of power draws for load disaggregation." [Online]. Available: http://arxiv.org/abs/1501.02954

[44] N. Batra *et al.*, "NILMTK: An open source toolkit for non-intrusive load monitoring," in *Proc. 5th Int. Conf. Future Energy Syst. (e-Energy)*, Cambridge, U.K., 2014, pp. 265–276.

BOCHAO ZHAO received the B.Eng. degree in 2014. He is currently pursuing the Ph.D. degree in electronic and electrical engineering with the University of Strathclyde, Glasgow, U.K. His research interests include graph signal processing and power disaggregation.

LINA STANKOVIC (M'04–SM'12) received the B.Eng. (Hons.) in electronic communications engineering and the Ph.D. degree from Lancaster University, in 1999 and 2003, respectively. From 2002 to 2007, she was with Lancaster University, first as a Research Associate, and then as a Lecturer. She has worked with BT Labs Martlesham Heath on digital video streaming, and Philips Research Eindhoven on signal acquisition and processing from 2-D optical discs. She is a Lecturer with the University of Strathclyde. She has authored four book chapters and 107 peer-reviewed research articles and was an Area Editor of the *International Journal of Electronics and Communications* (Elsevier) 2010-2015. Her main research areas lie in smart monitoring from sensor network platforms and meaningful information extraction that is user-centric, focused on algorithmic analysis of energy data to understand how people use appliances in the home and signal and image processing of biomedical data for motion assessment.

VLADIMIR STANKOVIC (M'03–SM'10) received the Dr.- Ing. (Ph.D.) degree from the University of Leipzig, Leipzig, Germany, in 2003. From 2003 to 2006, he was with Texas A&M University, College Station, first as a Research Associate and then as a Research Assistant Professor. From 2006 to 2007, he was with Lancaster University. Since 2007, he has been with the Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, U.K., where he is currently a Reader. He has co-authored four book chapters and over 160 peer-reviewed research papers.

He was an IET TPN Vision and Imaging Executive Team Member, an Associate Editor of the IEEE COMMUNICATIONS LETTERS, a member of the IEEE Communications Review Board, the Symposium Chair of the IEEE SmartGridComm-2013, and a Technical Program Committee Co-Chair of EUSIPCO-2012. He is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the Editor-at-Large of the IEEE TRANSACTIONS ON COMMUNICATIONS, and an Area Editor of *Elsevier Signal Processing: Image Communication*.

His research interests include energy disaggregation, signal information processing for smart homes and buildings, health monitoring, and user-experience driven image processing and communications.