PHASED: Phase-Aware Submodularity-Based Energy Disaggregation

Faisal M. Almutairi
University of Minnesota, Minneapolis, MN

Aritra Konar
University of Virginia, Charlottesville, VA

Ahmed S. Zamzam
National Renewable Energy Laboratory, Golden, CO

Nicholas D. Sidiropoulos
University of Virginia, Charlottesville, VA

ABSTRACT

Energy disaggregation is the task of discerning the energy consumption of individual appliances from aggregated measurements, which holds promise for understanding and reducing energy usage. In this paper, we propose PHASED, an optimization approach for energy disaggregation that has two key features: PHASED (i) exploits the structure of power distribution systems to make use of readily available measurements that are neglected by existing methods, and (ii) poses the problem as a minimization of a difference of submodular functions. We leverage this form by applying a discrete optimization variant of the majorization-minimization algorithm to iteratively minimize a sequence of global upper bounds of the cost function to obtain high-quality approximate solutions. PHASED improves the disaggregation accuracy of state-of-the-art models by up to 61% and achieves better prediction on heavy load appliances.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Theory of computation** → **Discrete optimization**; • **Hardware** → **Smart grid**.

KEYWORDS

Energy disaggregation, non-intrusive load monitoring, submodular functions, single (split)-phase connection, three-phase connection.

ACM Reference Format:
Faisal M. Almutairi, Aritra Konar, Ahmed S. Zamzam, and Nicholas D. Sidiropoulos. 2020. PHASED: Phase-Aware Submodularity-Based Energy Disaggregation . In *Proceedings of The 5th International Workshop on Non-Intrusive Load Monitoring (NILM '20), Nov 18, 2020, Virtual Event, Japan.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3427771.3427860

1 INTRODUCTION

Improving the energy efficiency of smart homes via machine learning (ML) methods constitutes an important research area with many potential benefits, such as reducing the adverse effects of energy consumption on the environment. Energy disaggregation/non-intrusive load monitoring (NILM) seeks to break down the energy usage of multiple household appliances from a single aggregated power measurement [13]. NILM benefits a plethora of applications in the areas of energy saving, automation in smart homes, anomaly detection, and life coaching and recommendations [23].

Many ML approaches have been proposed for NILM; see [10] and the references therein. Because the problem can be very ill posed, these methods are primarily supervised and require appliance-level training data available from homes with submeters (e.g., data summarized in [21]) for learning a model that generalizes to new (unseen) homes using only their aggregated power consumption. In this direction, sparse coding [8, 17, 20] and matrix/tensor factorization [4, 22, 25] approaches aim to learn a latent factor/dictionary from a training set, which is then used for disaggregation. The work in [17] proposed a customized dictionary learning method, where appliance-specific bases are learned from labeled training data such that the disaggregation error is minimized. Another approach in [22] used nonnegative matrix factorization (NMF), where one factor corresponds to the normalized appliance-level power consumption as the basis. The other factor forms the basis coefficients, which are constrained to add up to 1 for each appliance to impose the "groupin" effect. Although they are conceptually appealing, these methods require large training data to capture all possible appliance states, and they depend on the (hard to validate) assumption of common latent factors between the training and test sets. Neural network models have been deployed for the NILM task [16, 26]. For instance, the work in [26] proposed a network architecture, called sequence-to-point (seq2p), where the input is a window of the aggregated time series, and the output corresponds to the appliance power at the middle point in the given window. In addition to its large number of trainable parameters (> 30 M), the main drawback of seq2p is that it trains a separate model for each appliance independently; thus, it ignores the dependency among appliances (the aggregated signal is a joint function of all the constituent appliances).

Recently, the work in [1] demonstrated that the energy disaggregation can be posed as a constrained set-function maximization problem, which is NP–hard in its general form. The authors proposed a discrete block successive approximation algorithm that exploits the fact that the cost function is block-submodular [1]. Building on this line of work, we propose PHASED, a supervised framework for energy disaggregation that leverages the connectivity structure of the power distribution networks. To the best of our knowledge, there has not been *any* preexisting method that exploits such information. This allows us to obtain *multiple* aggregated measurements for each time instant, instead of a single measurement, thereby reducing the under-determinacy of the problem. Using appliance-level training data, we first learn to which energized line

NILM '20, Nov 18, 2020, Virtual Event, Japan

ACM ISBN 978-1-4503-8191-8/20/11…$15.00
https://doi.org/10.1145/3427771.3427860

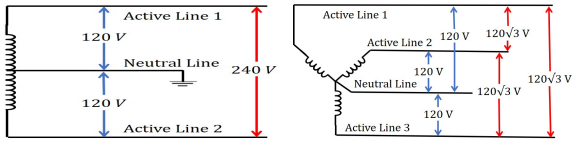

Figure 1: Split-phase (left) and three-phase (right).

(phase) an appliance is connected and the appliance consumption levels at its different states ('on', 'off', 'standby', etc.). The effectiveness of this model in breaking down aggregated signals is then evaluated on the test set. Although this requires solving a challenging, NP–hard, combinatorial optimization problem, we prove that the cost function can be decomposed as a *difference of submodular functions* (DSF)—an overview of submodular functions is provided in Appendix A. Leveraging the special properties of submodular functions [3, 11], we devise an efficient successive approximation algorithm for computing high-quality, albeit suboptimal solutions for the problem. In contrast to [1], we establish that the cost function can be expressed in DSF form over the *entire* time horizon, which results in a discrete approximation algorithm that features more attractive "all-at-once" updates[1]. PHASED improves the error of four distinct classes of state-of-the-art approaches by up to 61% when averaged over appliances.

2 PROBLEM STATEMENT

Given a household outfitted with L appliances, let $\{y_t\}_{t=1}^{T}$ represent the time series of the aggregated power consumption. The goal of energy disaggregation is to decompose y_t into L components of the form $y_t = \sum_{i=1}^{L} x_{i,t}$, where $x_{i,t}$ denotes the power consumption of appliance i at time t. A particularly challenging aspect of the problem is that it can be very under-determined because we wish to infer the power consumption of multiple appliances from a single measurement; however, in practice, the power distribution system supplying a household with electricity comprises multiple power lines, each corresponding to a different phase. The aggregated power consumption at a given instant, t, then comprises multiple measurements, $\{y_t^r\}_{r=1}^{R}$, where y_t^r is the power measured at the r^{th} line (wire) at time t, and $R \in \{2, 3\}$ is the number of lines depending on the low-voltage connection. The electrical networks usually employ one of the following two connections:

(i) **single-phase:** also known as *split-phase*, commonly used in North America for residential buildings. In this connection, the transformer takes a single-phase input and provides a 240-V output with a center tap that is connected to the ground, i.e., it provides 240-V that is divided into two 120-V live conductors. Light loads are connected between a live conductor and the neutral to receive 120-V, whereas heavy loads receive 240-V by being connected between two live conductors [9]—see Fig. 1 (left).

(ii) **three-phase:** common in commercial buildings in the United States and in residential buildings in Europe. In this connection, the power is delivered over three live conductors. The premises are fed with four lines (three live conductors and a neutral) [24]. Each live conductor corresponds to a single phase with a phase separation of 120° between any two live conductors. Fig. 1 (right) shows the three-phase wye connection.

[1]code is available at: https://github.com/FaisalAlmutairi/submodularity_based_NILM.

For both single- and three-phase-connected buildings, power consumption readings are often taken at every live conductor; however, the prevailing approaches in the literature do not consider the connectivity structure of the electrical feeder, and they make the simplifying assumption that aggregated power at a given time instant is drawn only from a single line, i.e., $R = 1$. Consequently, available information from other lines is summed, or even neglected.

3 PROPOSED METHOD: PHASED

Our proposed method, PHASED, is cognizant of the underlying residential feeder topology and exploits the readily available multiple aggregated power measurements—each corresponding to the power drawn from one of the lines supplying the household—to reduce the under-determinacy of the problem. Note that a particular appliance can be connected between either one of the live lines and the neutral, or between two live lines. Consequently, appliances that are connected to only one live line draw all of their consumed power from this particular line, whereas appliances connected between two live lines draw power from both lines. Formally:

$$y_t^r = \sum_{i=1}^{L} w_i^r x_{i,t}, \ \forall\, r \in [R], \tag{1}$$

If appliance i is connected to only a single line $r \in [R]$ (and the neutral), then $w_i^r = 1$ and $w_i^s = 0, \forall\, s \neq r$. Otherwise, if appliance i is connected between a pair of lines (r, s), then $0 < w_i^r < 1$, $0 < w_i^s < 1$, and $w_i^r + w_i^s = 1$.

We make the standard assumption that the power consumption profile of every appliance i can be approximated by a finite number $N_i \geq 2$ of *states* (i.e., operational modes). Let $\boldsymbol{\mu}_i \in \mathbb{R}_+^{N_i}$ denote a vector of the (approximately) constant power consumption levels of the i^{th} appliance over all of its states. Because each appliance can operate in only one state at a time, we can express the power consumed by appliance i at time t as:

$$x_{i,t} = \boldsymbol{\mu}_i^T \mathrm{e}_{i,t}, \ \forall\, i \in [L], \ t \in [T] \tag{2}$$

where $\mathrm{e}_{i,t} \in \{0, 1\}^{N_i}$ is a binary "selection" vector that represents the state of appliance i at time t and whose entries sum to 1.

3.1 Formulation

Conditioned on the power consumption profiles $\{\boldsymbol{\mu}_i\}_{i=1}^{L}$ and the connectivity weights $\{w_i^r\}_{(i,r=1)}^{(L,R)}$ being known *a priori*, the energy disaggregation problem boils down to choosing a state for each appliance at a time, t. Although exploiting the aggregated measurements from multiple lines somewhat reduces the ill-posedness of the problem, from an "equations versus unknowns" standpoint, it is always under-determined. Consequently, we exploit the fact that appliances change states infrequently over a short time horizon. Hence, we propose performing the energy disaggregation task over the entire time horizon while imposing temporal consistency on the evolution of the binary selection vectors. This leads to the following formulation:

$$\begin{aligned} \min_{\{\mathrm{e}_{i,t}\}_{(i,t=1)}^{(L,T)}} \ & \sum_{r,t=1}^{R,T} \left(y_t^r - \sum_{i=1}^{L} w_i^r \boldsymbol{\mu}_i^T \mathrm{e}_{i,t}\right)^2 - \sum_{i,t=1}^{L,T-1} \lambda_i \mathrm{e}_{i,t}{}^T \mathrm{e}_{i,t+1} \\ \text{s.t. } & \mathrm{e}_{i,t} \in \{0,1\}^{N_i}, 1^T \mathrm{e}_{i,t} = 1, \forall\, i \in [L], t \in [T] \end{aligned} \tag{3}$$

where the first term represents the least-squares data fit over all phases (lines); and the second term is a smoothness-inducing regularizer that seeks to maximize the similarity between the states of an appliance over consecutive time instants as in [1, 8]; and $\lambda_i \in \mathbb{R}_+$ is a regularization parameter (we set it to 1 in the experiments). The constraints in (3) guarantee the selection of only one state for each

appliance at a time. Evidently, this problem is a discrete quadratic program, which is NP–hard in its general form. As such, our objective is to design an approximation algorithm capable of yielding high-quality, albeit suboptimal solutions in polynomial time. As a first step, we equivalently reformulate (3) as a *subset selection* problem. This requires expressing (3) in set-notation, which is done as follows.

For each appliance $i \in [L]$, we define a "ground" set $\mathcal{A}_i := \{1, ..., N_i\}$ that represents the universe of states that appliance i can occupy. Then, let $\mathcal{S}_{i,t}$ be the singleton set that represents the state of appliance i at time t. Simple inspection reveals that $\mathrm{e}_{i,t}$ is the indicator vector of $\mathcal{S}_{i,t}$, i.e., $\mathrm{e}_{i,t} = \mathbb{1}_{\mathcal{S}_{i,t}}$. As an example, if appliance i has $N_i = 4$ states, and it is operating in the third state at t, then $\mathrm{e}_{i,t} = [0,0,1,0]^T \leftrightarrow \mathcal{S}_{i,t} = \{3\}$. To express the problem concisely, let the set $\mathcal{S}_t := \biguplus_{i=1}^{L} \mathcal{S}_{i,t}$ be the *disjoint* union of the sets $\{\mathcal{S}_{i,t}\}_{i=1}^{L}$, i.e., $\mathcal{S}_t$ "concatenates" the states of all appliances at t as $\mathcal{S}_t := [\mathcal{S}_{1,t}, \cdots, \mathcal{S}_{L,t}]$. Analogously, we define the set $\mathcal{T} := \biguplus_{i=1}^{L} \mathcal{A}_i$ to be the "super-universe" of all states across all appliances. Let $N := \sum_{i=1}^{L} N_i$. Then, we define:

$$\boldsymbol{\beta}^r := [w_1^r \boldsymbol{\mu}_1^T, w_2^r \boldsymbol{\mu}_2^T, \ldots, w_L^r \boldsymbol{\mu}_L^T]^T \in \mathbb{R}^N, \forall r \in [R] \tag{4}$$

which concatenates the consumption vectors of all appliances connected to line r *and* scales them by their respective connectivity weights, w_i^r. Next, define the matrix $\mathrm{B}^r := \boldsymbol{\beta}^r \boldsymbol{\beta}^{rT}$ and the vector $\mathrm{b}_t^r = 2y_t^r \boldsymbol{\beta}^r$ for each line $r \in [R]$. Finally, we define the diagonal matrix $\boldsymbol{\Lambda} := \mathrm{diag}(\lambda_1 1_{N_1}, \ldots, \lambda_L 1_{N_L})$, where 1_{N_1} is a vector of all ones of size N_i. Putting everything together and expanding the least-squares terms, (3) can be equivalently expressed as:

$$\min_{\{\mathcal{S}_t \in \mathcal{I}_t\}_{t=1}^T} \quad \sum_{r=1}^{R} \sum_{t=1}^{T} \left(\mathbb{1}_{\mathcal{S}_t}^T \mathrm{B}^r \mathbb{1}_{\mathcal{S}_t} - \mathbb{1}_{\mathcal{S}_t}^T \mathrm{b}_t^r \right) - \sum_{t=1}^{T-1} \left(\mathbb{1}_{\mathcal{S}_t}^T \Lambda \mathbb{1}_{\mathcal{S}_{t+1}} \right) \tag{5}$$

where the set $\mathcal{I}_t := \{\mathcal{S}_t \subset \mathcal{T} : |\mathcal{S}_t \cap \mathcal{A}_i| = 1, \forall i \in [L], t \in [T]\}$ guarantees that only one state is chosen per appliance at any time. To further simplify the problem representation, we define $\mathcal{S} := \biguplus_{t=1}^{T} \mathcal{S}_t$ as the set that contains the states of all appliances across all time instants. Note that $\mathcal{S} \subset \mathcal{V} := \biguplus_{t=1}^{T} \mathcal{T}$. We also define the block diagonal matrix $\mathrm{Q}^r := \mathrm{I}_T \otimes \mathrm{B}^r$, where I_T is the $T \times T$ identity matrix and $\otimes$ is the Kronecker product. Next, we define the time smoothness regularization matrix $\mathrm{R} := \mathrm{D} \otimes \boldsymbol{\Lambda}$, where $\mathrm{D} \in \mathbb{R}^{T \times T}$ is a symmetric Toeplitz matrix, whose first super- and sub-diagonal elements equal $1/2$, and the remaining entries are 0. Finally, let $\mathrm{b}^r := [\mathrm{b}_1^{rT}, \mathrm{b}_2^{rT}, \cdots, \mathrm{b}_t^{rT}]^T$. Armed with these definitions, we obtain the final subset-selection form of (3):

$$\min_{\mathcal{S} \in \mathcal{I}} \left\{ f(\mathcal{S}) := \sum_{r=1}^{R} \left(\mathbb{1}_{\mathcal{S}}^T \mathrm{Q}^r \mathbb{1}_{\mathcal{S}} - \mathbb{1}_{\mathcal{S}}^T \mathrm{b}^r \right) - \mathbb{1}_{\mathcal{S}}^T \mathrm{R} \mathbb{1}_{\mathcal{S}} \right\} \tag{6}$$

where $\mathcal{I} := \biguplus_{t=1}^{T} \mathcal{I}_t$. Although an exact minimization of the quadratic set functions is NP–hard in general, we now demonstrate that the cost function of (6) exhibits a special property that enables us to devise a simple polynomial-time approximation algorithm.

Proposition 1. *The set function $f(\mathcal{S})$ can be equivalently expressed as a DSF: $f(\mathcal{S}) = g(\mathcal{S}) - h(\mathcal{S})$, where $g(\mathcal{S}) := -\mathbb{1}_{\mathcal{S}}^T \mathrm{R} \mathbb{1}_{\mathcal{S}}$ and $h(\mathcal{S}) := \sum_{r=1}^{R} -\mathbb{1}_{\mathcal{S}}^T \mathrm{Q}^r \mathbb{1}_{\mathcal{S}} + \mathbb{1}_{\mathcal{S}}^T \mathrm{b}^r$ are submodular functions.*

3.2 Algorithm

To exploit the DSF form in our formulation, we utilize a discrete optimization analogue of the majorization-minimization (MM) procedure proposed in [14, 19]. The approach is iterative and consists of two main steps:

1) **Majorization:** At each iteration $k \in \mathbb{N}$, we compute a modular *upper* bound $u_{\mathcal{S}^k}^g(\mathcal{S})$ of $g(\mathcal{S})$ about the current solution set $\mathcal{S}^k$ that satisfies the following properties:

$$g(\mathcal{S}) \leq u_{\mathcal{S}^k}^g(\mathcal{S}), \forall \mathcal{S} \subset \mathcal{V}, \text{ and } g(\mathcal{S}^k) = u_{\mathcal{S}^k}^g(\mathcal{S}^k) \tag{7}$$

Similarly, a modular *lower* bound $v_{\mathcal{S}^k}^h(\mathcal{S})$ of $h(\mathcal{S})$ is constructed for the current solution set $\mathcal{S}^k$ such that:

$$h(\mathcal{S}) \geq v_{\mathcal{S}^k}^h(\mathcal{S}), \forall \mathcal{S} \subset \mathcal{V}, \text{ and } h(\mathcal{S}^k) = v_{\mathcal{S}^k}^h(\mathcal{S}^k). \tag{8}$$

2) **Minimization:** Upon replacing $g(\mathcal{S})$ by $u_{\mathcal{S}^k}^g(\mathcal{S})$ and $h(\mathcal{S})$ by $v_{\mathcal{S}^k}^h(\mathcal{S})$, we obtain a modular *upper* bound of $f(\mathcal{S})$, which is tight around the current solution set $\mathcal{S} = \mathcal{S}^k$. The resulting problem corresponds to minimizing a modular function

$$\min_{\mathcal{S} \in \mathcal{I}} m_k(\mathcal{S}) := u_{\mathcal{S}^k}^g(\mathcal{S}) - v_{\mathcal{S}^k}^h(\mathcal{S}) \tag{9}$$

which admits a simple solution. To see this, note that $m_k(\mathcal{S})$ is a modular function by construction, i.e., $m_k(\mathcal{S}) = \mathrm{m}_k^T \mathbb{1}_{\mathcal{S}}$. To compute the optimal solution, we simply inspect the entries of m_k corresponding to each subset $\mathcal{S}_{i,t}$ and pick the index of the smallest entry, $\forall i \in [L], t \in [T]$, which costs only $O(NT)$ in total.

Modular Upper Bound: Given a set $\mathcal{Y} \subseteq \mathcal{V}$, the super-differential set $\partial^g(\mathcal{Y})$ of a submodular function $g(\mathcal{Y})$ is defined as [15]: $\partial^g(\mathcal{Y}) = \{\mathrm{y} \in \mathbb{R}^n : g(\mathcal{X}) \leq g(\mathcal{Y}) + y(\mathcal{X}) - y(\mathcal{Y}), \forall \mathcal{X} \subseteq \mathcal{V}\}$, where every vector $\mathrm{y} \in \partial^g(\mathcal{Y})$ defines a modular function $y(\mathcal{X}) = \mathrm{y}^T \mathbb{1}_{\mathcal{X}}, \forall \mathcal{X} \subseteq \mathcal{V}$. A supergradient $\mathrm{y} \in \partial^h(\mathcal{Y})$ is used to define a modular upper bound function of the form: $u_{\mathcal{Y}}^g(\mathcal{X}) := g(\mathcal{Y}) + y(\mathcal{X}) - y(\mathcal{Y})$, which, by construction, satisfies the properties (7). A particular choice of a supergradient $\mathrm{u}_{\mathcal{Y}}^g \in \partial^g(\mathcal{Y})$ is given by [14]:

$$\mathrm{u}_{\mathcal{Y}}^g(j) = \begin{cases} g(\mathcal{Y}) - g(\mathcal{Y} \backslash \{j\}), & \forall j \in \mathcal{Y} \\ g(\{j\}) - g(\emptyset), & \forall j \notin \mathcal{Y} \end{cases} \tag{10}$$

With $\mathrm{u}_{\mathcal{Y}}^g$ obtained, we define the modular function for all subsets $\mathcal{S} \subseteq \mathcal{V}$ as $u_{\mathcal{Y}}^g(\mathcal{S}) = \mathbb{1}_{\mathcal{S}}^T \mathrm{u}_{\mathcal{Y}}^g$, which we then use as the desired upper bound function in the majorization step.

Modular Lower Bound: The subdifferential set of a submodular function $h(.)$ for a given set $\mathcal{Y} \subseteq \mathcal{V}$ is defined as [11, Section 6.2]: $\partial_h(\mathcal{Y}) = \{\mathrm{y} \in \mathbb{R}^n : h(\mathcal{X}) \geq h(\mathcal{Y}) + y(\mathcal{X}) - y(\mathcal{Y}), \forall \mathcal{X} \subseteq \mathcal{V}\}$. Let $\mathrm{v}_{\mathcal{Y}}^h \in \partial_h(\mathcal{Y})$ denote a subgradient of h at $\mathcal{Y}$. We need to compute such a subgradient for constructing our desired modular lower bound. To do so, it suffices to compute any element in the set of extreme points of $\partial h(\mathcal{Y})$, which can be exactly characterized by Theorem 6.11 in [11]. In [7], Edmonds presented a greedy procedure for computing such extreme points. Given a set $\mathcal{Y}$, let $\boldsymbol{\pi}$ be a permutation of the ground set $\mathcal{V} = [n]$, which maps the elements of $\mathcal{Y}$ to the first $|\mathcal{Y}|$ positions, i.e., $\boldsymbol{\pi}(i) \in \mathcal{Y}, \forall\, i \leq |\mathcal{Y}|$. The remaining $n - |\mathcal{Y}|$ positions of $\boldsymbol{\pi}$ can be assigned randomly. Every such permutation vector defines a chain of subsets $\mathcal{S}_{\boldsymbol{\pi}}^{(0)} \subset \mathcal{S}_{\boldsymbol{\pi}}^{(1)} \subset \ldots \subset \mathcal{S}_{\boldsymbol{\pi}}^{(n)}$ with elements $\mathcal{S}_{\boldsymbol{\pi}}^{(0)} = \emptyset$, and $\mathcal{S}_{\boldsymbol{\pi}}^{(i)} = \{\boldsymbol{\pi}(1), \boldsymbol{\pi}(2), \cdots, \boldsymbol{\pi}(i)\}, \forall\, i \in [n]$ ordered by inclusion, i.e., a (maximal) chain. Note that we have $\mathcal{S}_{\boldsymbol{\pi}}^{|\mathcal{Y}|} = \mathcal{Y}$. Using this chain, we define a vector $\mathrm{v}_{\mathcal{Y},\boldsymbol{\pi}}^h \in \mathbb{R}^n$:

$$\mathrm{v}_{\mathcal{Y},\boldsymbol{\pi}}^h(\boldsymbol{\pi}(i)) = \begin{cases} h(\mathcal{S}_{\boldsymbol{\pi}}^{(1)}) & \text{if } i = 1 \\ h(\mathcal{S}_{\boldsymbol{\pi}}^{(i)}) - h(\mathcal{S}_{\boldsymbol{\pi}}^{(i-1)}), & \text{otherwise} \end{cases} \tag{11}$$

By construction, $\mathrm{v}_{\mathcal{Y},\boldsymbol{\pi}}^h$ satisfies the description of an extreme point of $\partial_h(\mathcal{Y})$ in Theorem 6.11 in [11]. With vector $\mathrm{v}_{\mathcal{Y},\boldsymbol{\pi}}^h$ thus obtained, we define the modular function for all subsets $\mathcal{S} \subseteq \mathcal{V}$ as $v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S}) := \mathbb{1}_{\mathcal{S}}^T \mathrm{v}_{\mathcal{Y},\boldsymbol{\pi}}^h$. Further, it has been shown [12] that for every $\mathcal{Y} \subseteq \mathcal{V}$,

the modular function $v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{S})$ satisfies the following properties: (i) $v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{S}) \leq h(\mathcal{S}), \forall\, \mathcal{S} \subseteq \mathcal{V}$, and (ii) $v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{S}^{(i)}_{\boldsymbol{\pi}}) = h(\mathcal{S}^{(i)}_{\boldsymbol{\pi}}), \forall\, i \in [n]$. While (i) implies the lower bound property, (ii) implies that:

$$v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{S}^{|\mathcal{Y}|}_{\boldsymbol{\pi}}) = v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{Y}) = h(\mathcal{Y}). \tag{12}$$

Taken together, the obtained modular function $v^h_{\mathcal{Y},\boldsymbol{\pi}}(\mathcal{S})$ is a tight lower bound of the submodular function $h(\mathcal{Y})$ and satisfies the desired properties in (8). The PHASED algorithm is summarized in Algorithm 1 in Appendix B. The procedure exploits the DSF structure of the cost function to perform approximate minimization by successively minimizing a sequence of global upper bounds while respecting the constraints. The steps comprising the inner loop are computationally lightweight—refer to [2] for more details. Regarding the generated iterates, we have the following chain of inequalities: $f(\mathcal{S}^{k+1}) \leq f(\mathcal{S}^k) \cdots \leq f(\mathcal{S}^1)$; hence, PHASED monotonically reduces the cost function of (6).

Learning Connectivity and State Variables: The power profiles $\{\boldsymbol{\mu}_i\}_{i=1}^{L}$ are learned by performing the Lloyd-Max quantization on the power consumption sequence $\{x_i(t)\}_{t=1}^{T}$ in the training data, and setting $\boldsymbol{\mu}_i$ to be the centroid values of the quantization intervals. The number of quantization intervals equals the number of states, N_i, which is fixed beforehand. Thereafter, the connectivity weights, w_i^r, are obtained by solving the convex optimization problem:

$$\begin{aligned} \min_{\{w_i^r\}_{(i,r=1)}^{(L,R)}} \quad & \textstyle\sum_{r=1}^{R} \sum_{t=1}^{T} \left(y^r(t) - \sum_{i=1}^{L} w_i^r x_{i,t}\right)^2 \\ \text{s.t. } & 0 \leq w_i^r \leq 1,\ \textstyle\sum_{r=1}^{R} w_i^r = 1, \forall\, i \in [L], r \in [R] \end{aligned} \tag{13}$$

4 EXPERIMENTS

Datasets: We evaluate PHASED using two publicly available datasets: **REDD** and **ECO**. Each dataset represents one of the two power distribution systems described earlier as they were collected in homes on different continents. **REDD** [18] contains data from 6 homes in the United States (House 5 is omitted because it does not have enough data). The whole-home measurements consist of the power readings at two lines; hence, the structure of the distribution system is *split-phase*. **ECO** [6] contains data from 6 Swiss households (we omit House 3 because it does not have enough data after synchronizing the time series). The distribution system here is *three-phase*, and the aggregated power consumption of each household is available for each phase feeding the premises. For both datasets, we collect all the time-stamped readings that have both the aggregated *and* appliance-level measurements to ensure synchronized readings, then we down-sample to 1 reading/minute.

Baselines and metric: We compare PHASED to four quite different baselines to ensure broad evaluation. The baseline methods (explained in Section 1) are: (i) **DSC** (discriminative sparse coding) [17], (ii) **NMF** [22], (iii) **seq2p** [26], and (iv) **BSMA** (block successive modular approximation) [1]. We measure the percentage of energy deviated (*PED*) from the true consumption of appliance i in a house h at a time t using:

$$PED_i(t,h) := \frac{|x_i(t,h) - \hat{x}_i(t,h)|}{y(t,h)}, \tag{14}$$

where $x_i(t,h)$ and $\hat{x}_i(t,h)$ are the true and inferred power consumption for appliance i at time t in house h, and $y(t,h)$ is the aggregated power at t in h. Then, we present the average of *PED* (***APED***) over the total time ticks in all the houses:

Table 1: *APED*% of appliances in REDD and ECO (lower is better). Underline bold means best, bold is second best.

REDD						ECO					
Appliance	DSC	NMF	seq2p	BSMA	PHASED	Appliance	DSC	NMF	seq2p	BSMA	PHASED
Fridge	33.72	32.32	**16.71**	20.96	**20.17**	Fridge	21.53	**12.58**	**11.75**	14.00	13.85
Dishwasher	3.97	5.47	5.17	**2.98**	**2.22**	Dishwasher	5.63	16.85	18.91	**2.72**	**2.56**
Microwave	3.32	3.21	9.76	**3.12**	**2.84**	Microwave	12.78	15.21	**4.74**	7.03	**3.57**
Washer/dryer	10.23	13.93	**2.66**	**2.66**	**1.79**	Washer/dryer	30.53	**2.58**	3.21	2.84	**0.90**
Stove	4.94	4.46	**1.62**	4.02	**1.75**	Stove	2.11	1.65	7.41	**0.63**	**0.53**
AC	1.80	**1.57**	1.74	1.86	**1.64**	Freezer	26.74	22.31	**17.00**	**18.56**	25.06
Bathroom GFI	4.61	5.35	3.00	**1.01**	**0.71**	Work station	31.05	11.62	**3.00**	6.77	**6.68**
Outlet unknown	6.72	8.23	**2.63**	9.85	**4.94**	TV & stereo	17.45	12.33	**5.91**	16.28	**10.96**
Kitchen outlet	13.76	15.03	**5.43**	6.18	**5.33**	Tablets	19.50	12.25	0.47	**0.58**	**0.58**
Lighting	19.21	17.79	**5.53**	12.31	**9.12**	–	–	–	–	–	–
Average	10.23	10.73	**5.42**	6.49	**5.05**	**Average**	18.59	11.93	8.04	**7.71**	**7.19**

$$APED_i(t,h) := \frac{\sum_{t=1}^{T} \sum_{h=1}^{H} PED_i(t,h)}{\sum_{h=1}^{H} T_h} \tag{15}$$

where T_h is the length of the time series of house h. The essence of this metric is adopted from [4]. The percentage of energy correctly allocated [18] is a complementary measure that can be represented as $(1 - PED)$. We split the data for each home into two halves—one for training and the other for testing. Our approach and the BSMA baseline are optimization-based and do not require training a model; thus, the training data are used only to choose the state vectors, $\boldsymbol{\mu}_i$, the number of states, N_i, and the connectivity weights, w_i^r.

Results: Table 1 shows the prediction error for each appliance in the REDD and ECO data—we show appliances that appears three times or more. The homes in ECO do not have consistent types of appliances; thus, we also include the typical appliances (e.g., microwave, stove) in Table 1 in addition to the common ones among households. With the REDD data, PHASED has four appliances with the *APED* less than 2%, whereas all the baselines have only two appliances less than 2%. Compared to the baselines, PHASED significantly improves the prediction of appliances. PHASED reduces the average of the *APED* among all appliances with DSC, NMF, seq2p, and BSMA by 50.6%, 52.9%, 6.88%, and 22.2%, respectively, on the REDD data. PHASED also improves the mean of the *APED* among DSC, NMF, seq2p, and BSMA using the ECO data by 61.34%, 39.75, 10.63%, and 6.79%, respectively. Moreover, PHASED has the best (or comparable in a few cases) performance for appliances with heavier load (e.g., washer/dryer, AC, fridge, and stove) and appliances with flexible usage time, e.g., dishwasher. Note that in a recent survey study [5], seq2p has been shown to be the strongest baselines with heavy load appliances. For instance, the *APED* of PHASED with washer/dryer in the REDD data is only 17.45%, 12.83%, 67.16%, and 67.24% of the *APED* of DSC, NMF, seq2p, and BSMA, respectively. With the ECO data, the washer/dryer error percentage of our method is only 2.96%, 34.98%, 28.13%, and 31.85% of the *APED* of the DSC, NMF, seq2p, and BSMA, respectively.

5 CONCLUSIONS

We presented a supervised framework for energy disaggergation that exploits the structure of power distribution systems by using multiple aggregated measurements to improve the disaggregation accuracy. The proposed approach formulates the problem as minimizing the difference between two submodular functions, subject to combinatorial constraints. Leveraging this form, we devised an iterative approximation algorithm that minimizes a sequence of global modular upper bounds on the cost function. The algorithm provably exhibits a non-increasing cost and features computionally lightweight updates. The effectiveness of PHASED was shown against four state-of-the-art baselines on two datasets with different power connectivity structures.

ACKNOWLEDGMENTS

The work of A. Konar and N. D. Sidiropoulos was supported in part by the National Science Foundation under Grant NSF IIS-1908070. This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. The work of A. S. Zamzam was supported in part by the Laboratory Directed Research and Development Program at the National Renewable Energy Laboratory. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

A OVERVIEW OF SUBMODULAR FUNCTIONS

Given a ground set of n elements, $\mathcal{V} := \{v_1, \cdots, v_n\}$, consider the set function $f : 2^{\mathcal{V}} \to \mathbb{R}$ that assigns a real value to any subset $\mathcal{S} \subseteq \mathcal{V}$. Among the set functions, the subclass of the *submodular* functions is notable for exhibiting many properties similar to both convex and concave functions, and it arises in many applications in machine learning [3]. Formally, a set function, $f(.)$, is submodular if and only if it satisfies $f(\mathcal{X} \cup \{v\}) - f(\mathcal{X}) \geq f(\mathcal{Y} \cup \{v\}) - f(\mathcal{Y})$ for all $\mathcal{X} \subseteq \mathcal{Y} \subseteq \mathcal{V} \backslash \{v\}$. That is, given any subset of elements $\mathcal{X}$, the marginal gain derived by adding an element v to $\mathcal{X}$ does not increase when we instead add v to the superset $\mathcal{Y}$. Hence, submodular functions exhibit a natural diminishing returns property. A submodular set function $y(.)$ is said to be *modular* if and only if there exists a vector $\mathrm{y} \in \mathbb{R}^n$ for all subsets $\mathcal{X} \subseteq \mathcal{V}$ such that $y(\mathcal{X}) = \mathrm{y}^T \mathbb{1}_{\mathcal{X}} = \sum_{e \in \mathcal{X}} \mathrm{y}(e)$.

B PHASED ALGORITHM

We present the summary of the algorithmic procedure in Algorithm 1 below—code is available here.

Algorithm 1 : PHASED Algorithm

Initialization: Set $k := 0$, $\mathcal{S}^0 \in \mathcal{I}$ (randomly initialization).

Repeat: 1) Generate permutation $\boldsymbol{\pi}$ using $\mathcal{S}^k$

2) Compute modular upper bound $u^g_{\mathcal{S}^k}(.)$ of $g(.)$ using (10)

3) Compute modular lower bound $v^h_{\mathcal{S}^k, \boldsymbol{\pi}}(.)$ of $h(.)$ using (11)

4) Compute $\mathcal{S}^{k+1} \in \arg\min_{\mathcal{S}_t \in \mathcal{I}} m_k(\mathcal{S})$ in (9) via linear scan

5) Set $k := k + 1$.

Until stopping criterion is met

REFERENCES

[1] Faisal M Almutairi, Aritra Konar, and Nicholas D Sidiropoulos. 2018. Scalable Energy Disaggregation Via Successive Submodular Approximation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, AB, Canada, 2676–2680.

[2] Faisal M. Almutairi, Aritra Konar, Ahmed S. Zamzam, and Nicholas D. Sidiropoulos. 2020. PHASED: Phase-Aware Submodularity-Based Energy Disaggregation. *arXiv preprint arXiv:2010.00696* (2020).

[3] Francis Bach. 2013. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning* 6, 2-3 (2013), 145–373.

[4] Nipun Batra, Yiling Jia, Hongning Wang, and Kamin Whitehouse. 2018. Transferring decomposed tensors for scalable energy breakdown across regions. In *Proceddings of the 32nd AAAI Conference on Artificial Intelligence*. New Orleans, Louisiana, United States, 240–247.

[5] Nipun Batra, Rithwik Kukunuri, Ayush Pandey, Raktim Malakar, Rajat Kumar, Odysseas Krystalakos, Mingjun Zhong, Paulo Meira, and Oliver Parson. 2019. Towards reproducible state-of-the-art energy disaggregation. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. New York, NY, United States, 193–202.

[6] Christian Beckel, Wilhelm Kleiminger, Romano Cicchetti, Thorsten Staake, and Silvia Santini. 2014. The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms. In *Proceedings of the 1st ACM International Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys 2014)*. Memphis, TN, United States, 80–89.

[7] Jack Edmonds. 1970. Submodular functions, matroids, and certain polyhedra. *Edited by G. Goos, J. Hartmanis, and J. van Leeuwen* 11 (1970).

[8] Ehsan Elhamifar and Shankar Sastry. 2015. Energy Disaggregation via Learning 'Powerlets' and Sparse Coding. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. Austin, Texas, United States, 629–635.

[9] Mohamed S ElNozahy and Magdy MA Salama. 2013. A comprehensive study of the impacts of PHEVs on residential distribution networks. *IEEE Trans. Sustainable Energy* 5, 1 (2013), 332–342.

[10] Anthony Faustine, Nerey Henry Mvungi, Shubi Kaijage, and Kisangiri Michael. 2017. A survey on non-intrusive load monitoring methodies and techniques for energy disaggregation problem. *arXiv preprint arXiv:1703.00785* (2017).

[11] Satoru Fujishige. 2005. *Submodular functions and optimization* (2nd ed.). Annals of Discrete Mathematics, Vol. 58. Elsevier.

[12] M. Grötschel, L. Lovász, and A. Schrijver. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* 1, 2 (June 1981), 169–197.

[13] George William Hart. 1992. Nonintrusive appliance load monitoring. *Proc. IEEE* 80, 12 (1992), 1870–1891.

[14] Rishabh Iyer and Jeff Bilmes. 2012. Algorithms for Approximate Minimization of the Difference Between Submodular Functions, with Applications. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*. Catalina Island, CA, 407–417.

[15] Rishabh Iyer, Stefanie Jegelka, and Jeff Bilmes. 2013. Fast semidifferential-based submodular function optimization: Extended version. In *Proceedings of the International Conference on Machine Learning (ICML)*. Atlanta, GA, United States.

[16] Jack Kelly and William Knottenbelt. 2015. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. Seoul, South Korea, 55–64.

[17] J Zico Kolter, Siddharth Batra, and Andrew Y Ng. 2010. Energy disaggregation via discriminative sparse coding. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*. Vancouver, British Columbia, Canada, 1153–1161.

[18] J Zico Kolter and Matthew J Johnson. 2011. REDD: A public data set for energy disaggregation research. In *Workshop on Data Mining Applications in Sustainability (SustKDD) in the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. San Diego, CA, United States, 59–62.

[19] Mukund Narasimhan and Jeff A Bilmes. 2005. A submodular-supermodular procedure with applications to discriminative structure learning. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*. Edinburgh, Scotland, 404–412.

[20] Shalini Pandey and George Karypis. 2019. Structured Dictionary Learning for Energy Disaggregation. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems*. Phoenix, AZ, United States, 24–34.

[21] Oliver Parson, Grant Fisher, April Hersey, Nipun Batra, Jack Kelly, Amarjeet Singh, William Knottenbelt, and Alex Rogers. 2015. Dataport and NILMTK: A building data set designed for non-intrusive load monitoring. In *Proceedings of the IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Orlando, Florida, United States, 210–214.

[22] Alireza Rahimpour, Hairong Qi, David Fugate, and Teja Kuruganti. 2017. Non-intrusive energy disaggregation using non-negative matrix factorization with sum-to-k constraint. *IEEE Transactions on Power Systems* 32, 6 (2017), 4430–4441.

[23] Changho Shin, Seungeun Rho, Hyoseop Lee, and Wonjong Rhee. 2019. Data requirements for applying machine learning to energy disaggregation. *Energies* 12, 9 (2019), 1–19.

[24] Theodore Wildi. 2002. Electrical machines, drives, and power systems. *New Jersey: Upper Saddle River* (2002).

[25] Ahmed S Zamzam, Bo Yang, and Nicholas D Sidiropoulos. 2020. GRATE: Granular Recovery of Aggregated Tensor Data by Example. *arXiv preprint arXiv:2003.12666* (2020).

[26] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. 2018. Sequence-to-point learning with neural networks for non-intrusive load monitoring. In *Proceddings of the 32nd AAAI conference on artificial intelligence*. New Orleans, Louisiana, United States.