Time Series Clustering for Demand Response
An Online Algorithmic Approach

Ranjan Pal, Student Member, IEEE, Charalampos Chelmis, Member, IEEE, Marc Frincu, Member, IEEE, Viktor Prasanna, Fellow, IEEE

Abstract

The widespread monitoring of electricity consumption due to increasingly pervasive deployment of networked sensors in urban environments has resulted in an unprecedentedly large volume of data being collected. To improve sustainability in Smart Grids, real-time data analytics challenges induced by high volume and high dimensional context-based data need to be addressed. Particularly, with the emerging Smart Grid technologies becoming more ubiquitous, analytics for discovering the underlying structure of high dimensional time series data are crucial to convert the massive amount of fine-grained energy information gathered from residential smart meters into appropriate Demand Response (DR) insights. In this paper, we propose an online time series clustering approach to systematically and efficiently manage the energy consumption data deluge, and also capture specific behavior i.e., identify households with similar lifestyle patterns. Customers can in this way be segmented into several groups that can be effectively used to enhance DR policies for real time automatic control in the cyberphysical Smart Grid system. Due to the inherent intractability of the ‘optimal clustering’ problem, we propose a novel randomized approximation clustering scheme of electricity consumption data, aiming at addressing three major issues: (i) designing a resource-constrained, online clustering technique for high volume, high dimensional time series data (ii) determining the optimal number of clusters that gives the best approximate clustering configuration, and (iii) providing strong clustering performance guarantees. By the term ‘performance guarantees’, we imply algorithm performance with respect to the best clustering possible for the given data. Our proposed online clustering algorithm is time efficient, achieves a clustering configuration that is optimal within provable worst case approximation factors, scales to large data sets, and is extensible to parallel and distributed architectures. The applicability of our algorithm goes beyond that of the Smart Grid and includes any scenario where clustering needs to be done on high volume and in real-time under space and time constraints.

Keywords
time series, online clustering algorithm, real-time analytics, Smart Grid, demand response

I. Introduction

The emergence of Cyber-Physical Systems (CPS) such as smart grids is leading to extensive digital sampling of power networks [42]. This has been possible due to the pervasive deployment of sensing instruments that generate large volumes of data with high velocity. The growing prevalence of smart meters deployed in large numbers by US utilities for example, has enabled the collection of energy consumption data from residential and commercial customers at unprecedented scales [37]. Specifically, energy consumption is sampled and reported back to the utility at 15-min granularity leading to a 3,000× jump in the volume of data traditionally collected by power utilities [39]. Such growing availability of energy consumption data offers unique opportunities in understanding the dynamics on both sides of the meter; customer behavior on the consumption side and operating requirements, planning, and optimization on the utility side. The data deluge however imposes utilities with the challenging tasks of managing massive sets of fine-grained electricity consumption data and applying data analytics to perform data-driven coordination of energy resources and efficiently deal with peak demand (e.g., by Demand Response (DR) programs [30].) The high dimensionality of electricity consumption data in particular makes traditional machine learning and data mining techniques computationally intractable [34]. In this introductory section we first introduce briefly the demand response mechanism as a representative Smart Grid application, and explain why the related ‘predictive analytics of energy consumption’ problem is important but at the same time intractable and challenging. We then state our proposed research contributions.

A. Demand Response in a Nutshell

With the emerging Advanced Metering Infrastructure (AMI) becoming more ubiquitous, unique opportunities for intelligent management of the power grid arise in order to improve efficiency and reliability, mainly due to the ability of extracting many features from electricity consumption shapes. One canonical Smart Grid application where such feature information is useful is Demand Response (DR). In this application, the utility uses power consumption data (e.g., peak usage, duration and time of...
day, etc.) at a micro time scale from individual customers in the service area to forecast the future demand and initiate energy curtailment programs that can avoid a supply-demand mismatch. A step further is the Dynamic DR (D^2R) which moves away from the static decision making allowing utilities to dynamically choose when and how to shed consumption often in real-time with a latency of few seconds. These curtailment programs target individual customers whose power usage is expected to increase in the near future, and in turn offers them incentives to shift their impending demand in a bid to relieve the stress on the power grid. In addition to energy consumption curtailment, predicting consumption demand of consumers for the next few hours helps a utility plan for additional generation, say, by using backed up stored energy, or purchasing power from the energy market.

B. Challenges to DR Implementation

Although effective demand response promises finding “appropriate” customers for the right incentive program, due to heterogeneity in customer behavior, the implementation of such programs can turn out to be very costly. Modeling each user separately has advantages (e.g., personalized models may capture rich, individual characteristics leading to more accurate forecasts), however the computational resources required to process a massive data sets would be prohibitive for most utilities operating in the public sector as the data generated by sampling a large population at 15-minute intervals is enormous. For example, electricity consumption values collected at 15-minute intervals by large utilities such as the Los Angeles Department of Water and Power that has over a million consumers [39] amounts to \( \approx 42 \) billion entries per year.

While parallel techniques (e.g., MPI, MapReduce) could be used to model each individual in parallel finding the best model requires \( \approx 54s \) per customer in an ideal parallel scenario [19] if 6 different models are required. Combined with the latency needed to get the data and make a decision we obtain a duration that is too long for dynamic demand response (D^2R) real-time (in the order of few seconds) decisions.

Further, customers vary widely in their energy usage patterns causing prediction models to have high errors. Aggregating data from multiple customers into “virtual consumers” instead, reduces the variability of each virtual customer, and thus reduces the prediction error of the aggregated load of each cluster of customers [37, 39]. Intuitively, it is expected that people sharing certain characteristics (e.g., similar lifestyles or household appliances) will generally exhibit similarities in their consumption behavior. This hypothesis has been hard to validate in the past because of insufficient consumption data. With the advent of smart meters however it has become possible to analyze at scale, finely granular consumption at the individual and group level. More importantly, splitting users in different groups by characterizing individual consumption facilitates the application/recommendation of diverse consumption curtailment strategies across the user base.

C. High-Dimensional Time Series Clustering

Research Motivation: Previous efforts on customer segmentation for DR have resorted to the well known unsupervised learning technique of clustering as their primary tool, because of its simplicity and effectiveness. These efforts have reported interesting findings [13, 16, 17, 41, 24, 25] (refer to Section 2.), but did not analyze the properties of their algorithms nor did they provide guarantees on the quality of the clustering results. Most clustering techniques in the above mentioned efforts depend heavily on initial seeding mechanisms which might drive these mentioned algorithms to output bad segmentations [35].

Solution Insight: We propose a methodology to systematically form virtual customers in Smart Grids in a provably optimal and efficient manner. The notion of efficiency is with respect to both computational time and space. Specifically, we observe \( N \) high dimensional time series \( X_n, n = 1, \ldots, N \), each representing the energy consumption of customer \( n \) at regular intervals \( \Delta t \) (here every 15-minutes), which we intent to cluster according to similarities in usage patterns. The high dimensionality of the time series data is a direct result of the granularity at which a smart meter reports energy consumption. For consumption values been collected every 15 minutes, a daily observation of 15-min meter data consists of 96 dimensions. Formally, a daily energy consumption time series is defined as \( c = \{c_t\}, \quad t = 1, \ldots, 96 \), where \( c_t \) is the energy in kWh consumed in the \( t \)-th 15-min interval. For consumption values been collected every 1 minute, the length of \( c \) increases to 1,440 dimensions. As AMI technology enables transmission of even finer-grained consumption data (e.g., every second), understanding consumption behavior requires appropriate statistical analysis to be used. More importantly, such approaches need to be scalable with the number of smart meters and granularity, as well as time ans space efficient to constitute computationally viable solutions.

We assume that the compute environment has limited memory which cannot store the entire large-scale, high-dimensional time series data set obtained from smart meter readings. Thus, our objective is to divide all time series data points into zones of similar data points in the best possible online manner (see Section 4.) and in the fastest possible way, where the time component wise sum of the data points in each zone forms the virtual consumer data point for that zone.

Finding \( K \) representative groups that yield the least within-cluster distance is a standard clustering problem. The \textit{K}-means algorithm is a popular statistical clustering approach and as such had been extensively applied for time series clustering [13, 24, 25]. However, the number of clusters needs to be predetermined in the classical \textit{K}-means algorithm. As it is hard to set an appropriate value for \( K \) without proper reasoning on the grounds of adopting a certain value, adaptive \textit{K}-means algorithms [25] have been proposed. Even though the traditional clustering problem is computationally intractable [15], we posit that it is feasible to organize a massive corpus of high dimensional time series data into clusters near optimally, in polynomial time. Specifically, we propose an unsupervised method to calculate the optimal clustering configuration for the given data set within a constant factor approximation gap under computationally constrained environments.
D. Research Contributions

Our main contributions are theoretical in nature and specifically suited to use cases geared towards effective Smart Grid demand response. However, our contributions are extendible to multiple scenarios where clustering needs to be done on high volume data, in real-time fashion, and under joint resource constraints of time and space.

- We design and analyze a batch randomized approximation algorithm, that clusters a given set of high dimensional smart meter time series data points in polynomial time. For any given data set, our algorithmic output approximates a clustering configuration that is within a constant factor gap, i.e., $O(1)$ gap, of the optimal clustering configuration. (see Section 4.)
- Using the above-mentioned batch randomized approximation algorithm, we propose and analyze an online randomized approximation algorithm that clusters a given set of potentially high dimensional time series data points into $k$ pre-specified clusters in polynomial time. Our algorithm can be deployed on a commodity machine with minimal memory requirements. More specifically, the space complexity to store $n$ very high dimensional data points, where $n$ is in the order of millions for a large metropolitan locality, is large. As a result, we might not have the entire data set at our disposal for cluster computations, and the main memory needs to fetch blocks of time series data from time to time and provide them as inputs to our algorithm that in turn appropriately updates the clustering configuration. For any given data set, our proposed online clustering algorithm is based on a divide-and-conquer approach and approximates a clustering configuration that is within $O(\log k)$ gap of the optimal clustering configuration, where $k$ is the pre-specified number of clusters. (see Section 5.)
- We extend existing literature on time series clustering in Smart Grids to give provable theoretical guarantees to the customer segmentation problem using clustering, for which only heuristic solutions are known till date (refer to Section 2.).

The rest of the paper is organized as follows - In Section 2, we review related work on predictive analysis of time series data. We discuss our problem setting and model preliminaries in Section 3. In Sections 4 and 5, we propose and analyse our algorithms for batch and online clustering. In Section 6, we present early experimental results regarding the practical utility of our approach. We conclude our paper in Section 7.

II. RELATED WORK

Recent years have seen an increasing interest in energy consumption, motivated by both energy markets and environmental aspects. Of particular emphasis has been accurate prediction models of energy usage, as one of the key challenges in Smart Grid is to reliably, accurately and efficiently predict future supply and demand trends [37, 11] to allow utility providers and consumers to prepare for peak periods as well as to plan for DR activities. Research on electricity demand forecasting considers longterm and medium-term prediction for utility planning and maintenance purposes, and short-term forecast for economic scheduling [3]. As utilities move towards $D^2R$, very short-term predictions are required for real-time control.

Currently, US utilities have deployed millions of smart meters that collect energy consumption data from residential and commercial customers [38]. Such growing availability of energy consumption data offers unique opportunities in designing segmentation strategies of household energy use to support demand response and energy efficiency programs design and customer targeting [33]. The introduction of smart meters has driven studies on high resolution time series modeling and customer clustering [13, 16, 17, 41, 24, 25]. For example, self-organizing maps (SOM) and $k$-means are used for load pattern mining [17]. A variety of clustering algorithms to segment consumers with similar consumption behavior have been examined in [13]. More closely related to our methodology, [24, 25] decompose daily electricity usage time series into representative load shapes by utilizing adaptive $k$ means and summarized utilizing hierarchical clustering. The final outcome of most consumer segmentation techniques is sensitive to initial seeding mechanisms. More importantly, to the best of our knowledge, none of the prior work has thus far provided performance guarantees with respect to the optimal segmentation possible [26].

Extensive literature exists on clustering of time series data [6, 26]. Typical computational issues include achieving the optimal clustering configuration, which is computationally intractable, i.e., NP-Hard [15]. Static techniques such as relocation and agglomerative hierarchical clustering have been applied on time series data, but as the number of dimensions grows with the length of the time series, these techniques become computationally expensive [26]. Feature-based techniques, such as wavelets [27], attempt to reduce the problem space by extracting motifs from the time series and replacing the time series with (a much smaller number of) motifs, thus achieving dimensionality reduction.

An exact $k$-means algorithm [21] has running time $O(n^{kd})$, where $d$ is the dimensionality of the data. Numerous polynomial time approximation schemes [18, 23, 31, 36] have been proposed. Such schemes are however highly exponential (or worse) in $k$ and therefore impractical even for relatively small values of $n$, $k$, and $d$. The most popular $k$-means clustering algorithm [29] is known to converge to a local optima. Motivated by the popularity of this algorithm in a variety of scientific and industrial applications [6], we base our proposed solution on Lloyd’s algorithm. Kanungo et al. [22] proposed an $O(n^3e^{-d})$ algorithm that is $(9+\epsilon)$ competitive. However, $n^3$ compares unfavorably with the almost linear running time of Lloyd’s method, and the exponential dependence on $d$ can also be problematic. A combination of Kanungo et al. and Lloyd’s algorithm has also been proposed, but in order to avoid the exponential dependence on $d$, this approach sacrifices all approximation guarantees. A constant probability $O(1)$ approximation with running time of $O(nkd)$ (but only if $k$ is sufficiently large and the spread is sufficiently small) was achieved using successive sampling [32]. Other $O(1)$ approximation algorithms assume that data is independent and
identically distributed (i.i.d.) [14], or that the data can be clustered with well separated means [12, 35]. Finally, an $O(nkd)$ algorithm for generic data sets that is $O(\log k)$ competitive was proposed in [5].

The main drawback of the above mentioned techniques is their inability to operate under memory limitations. For compute environments with limited memory constraints, simple to implement online heuristics that process small batches of data or one data object in each iteration have been proposed [28, 39]. Such approaches however provide no guarantees on the optimality gap of their produced clustering configurations. Our proposed randomized online clustering algorithm leverages the work of [28, 39, 1, 5, 20], while at the same time providing provable worst case performance guarantees with respect to the optimality of cluster configuration. Specifically, (i) our online clustering algorithm is running in polynomial time, and (ii) it provides a cluster configuration that is within $O(\log k)$ factor of the optimal clustering configuration.

III. PROBLEM SETTING AND PRELIMINARIES

In this section, we first state our problem setting followed by a description of the model preliminaries.

A. Problem Setting

We consider the setting where a fixed, large number of customers in a metropolitan locality exist, whose time series data of energy consumption for a given period of time, e.g., a month time granularized into 15 minute intervals, is known to the local utility. The utility wants to utilize each consumer’s data to make energy consumption predictions with the goal to optimize DR. As mentioned in Sections 1 and 2, the utility wants to adopt an aggregate clustering technique, whereby virtual consumers are formed from the given data set, where each virtual consumer represents a cluster, and the time series data for the virtual consumer is the component wise aggregate of the time series data of each consumer within the cluster [39]. The goal is to minimize uncertainty induced by variability of predictions for individual customers. We are therefore interested in minimizing the cumulative consumption prediction error by predicting energy consumption of virtual consumers.

The utility needs to take care of the following important aspect regarding regressive predictions at fine grained intervals: (i) the data per consumer is of high dimensions, and (ii) the utility has limited memory (much smaller than the the number of data points) to take into account all the data in one go in order to perform cluster formation computations. On the other hand the cluster formation process should be optimal, fast, and practically light weight, and as a result the utility can at most afford to make a constant number of passes of the data as per its memory limitations, and execute computations satisfying speed requirements. Only after the clustering process with existing data is completed can consumer aggregation and time series prediction methods can be used.

One could argue here that with the power of modern hardware and advances in distributed computing, memory and computational power will not be an issue for performing smart grid analytics and in supporting general cyber-physical systems (CPS) services in general. While this may be true, it is also evident that (i) the rate of increase in the data deluge with cyber-physical systems becoming more and more popular in our day-to-day lives, might just be greater than the pace at which advanced hardware can smoothly support corresponding data volumes, (ii) certain small/medium scale organizations providing smart grid services (e.g., microgrids) may not have the economic advantages to possess advanced hardware, and (iii) computing services that co-locate many tenants (utilities) providing CPS services (in our case smart grid related computing services) and provide utilities computing resources may have aggregate workloads that exceed their current capacities. Our proposed model setting targets such use cases.

B. Model Preliminaries

The $k$-means clustering problem is defined as follows: Given $n$ points $X \subseteq \mathbb{R}^d$ and a weight function $w : X \rightarrow \mathbb{R}^1$, the goal is to find a subset $C \subseteq \mathbb{R}^d, |C| = k$ such that the following quantity is minimized: $\phi_C = \sum_{x \in X} w(x)D(x, C)^2$, where $D(x, C)$ denotes the $l_2$ distance of $x$ to the nearest point in $C$. When the subset $C$ is clear from the context, we denote this distance by $D(x)$. In our problem context, each point is a time series of energy consumption values for a consumer. The dimension of each data point is the number of intervals for which the time series information is recorded. The number of intervals are assumed to be known in advance and is homogenous for all data points. Also, for two points $x, y, D(x, y)$ denotes the $l_2$ distance between $x$ and $y$. The subset $C$ is alternatively called a clustering of $X$ and $\phi_C$ is called the potential function corresponding to the clustering. We will use the term center to refer to any $c \in C$.

We now state some definitions related to the algorithms of clustering which will be used throughout the paper.

(Competitive ratio, $b$-approximation). Given an algorithm $B$ for the $k$-means problems, let $\phi_C$ be the potential of the clustering $C$ returned by $B$ (on some input set which is implicit) and let $\phi_{C_{OPT}}$ denote the potential of the optimal clustering $C_{OPT}$. Then the competitive ratio is defined to be the worst case ratio $\phi_C$. The algorithm $B$ is said to be $b$-approximation algorithm if $\frac{\phi_C}{\phi_{C_{OPT}}} \leq b$.

1For unweighted cases, $w(x) = 1$ for all $x \in X$
The previous definition might be too strong for an approximation algorithm for some purposes. For example, the clustering algorithm performs poorly when it is constrained to output \( k \) centers but it might become competitive when it is allowed to output more centers. In order to relax the strength of the definition, we state the following alternative definition.

\((a,b)\)-approximation - We call a clustering algorithm \( B \), \((a,b)\)-approximation for the \( k \)-means problem if it outputs a clustering \( C \) with \( ak \) centers with potential \( \phi_{C} \) such that \( \frac{\phi_{C}}{\phi_{OPT}} \leq b \) in the worst case. Where \( a > 1, b > 1 \).

Online Algorithm - We adopt the flavor of the traditional definition of an online algorithm in computer science literature, i.e., the algorithm is unaware of the data size and the order of arrival of data, but change the definition a little for the purposes of this paper, without violating any fundamental aspects of an online algorithm. More specifically, we call an algorithm online if for a given data set, (i) the size of the data set is significantly larger than the memory available to hold the data set, (ii) the algorithm divides the whole data set into parts\(^2\) and each part (unseen by the algorithm before) arrives as input to the algorithm in stages, (iii) the algorithm makes a single scan of each part, does some intermediary computations on the part, but does not store that the part at the end of the scan, and (iv) does final computations at the end of seeing the whole data set by accounting for the intermediary computations done earlier for each part.

IV. COMPETITIVE BATCH CLUSTERING

In this section we design a batch clustering algorithm, BTSC, that is based on the \( O(\log k) \)-competitive \( k \)-means algorithm in [5] but generates a clustering configuration that is \( O(1) \)-competitive but with potentially \( ak \) number of clusters, for a given \( k \) (i.e., we have an \( ak \)-means algorithm), where \( a > 1 \). We will use this algorithm in Section 5 to develop an online \( k \)-means clustering algorithm. The first part of this section describes the algorithm, and the second part analyzes the algorithm.

A. The Algorithm

Our algorithm is very similar to the one in [5], except that instead of picking one center in every round we choose \( O(\log k) \) centers.

Algorithm 1: BTSC finds a batch clustering

\[
\text{Input:} \text{ (a) Point Set } X \subset \mathbb{R}^d \text{ of consumer time series data. Let } n = |X|, \text{ and (b) Number of desired clusters, } k \in \mathbb{N}. \\
\text{Output:} \text{ A clustering configuration, } C \\
1 \text{ Choose } 3\log k \text{ centers independently and uniformly at random from } X \\
2 \text{ Repeat } k-1 \text{ times} \\
3 \quad \text{ Choose } 3\log k \text{ centers independently w.p. } \frac{D(x)^2}{\sum_{x \in X} D(x)^2} \\
4 \quad \text{ for } i \leftarrow 1 \text{ to } k \text{ do} \\
5 \quad \quad \text{ Set the cluster } C_i \text{ to be the set of points in } X \text{ that are closer to } c_i \text{ that they are to } c_j \text{ for all } j \neq i \\
6 \quad \text{ for } i \leftarrow 1 \text{ to } k \text{ do} \\
7 \quad \quad \text{ Set } c_i \text{ to be the center of mass of all points in } C_i \text{ such that } c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \\
8 \text{ Repeat Steps 4 to 7 until } C \text{ no longer changes} \\
9 \text{return } C
\]

B. Algorithmic Analysis

Before we analyse BTSC, we will need two results from [5]. We first state these results.

Lemma 1. Let \( A \) be an arbitrary cluster in \( C_{OPT} \), and let \( C \) be the clustering with just one center, chosen uniformly at random from \( A \). Then (i) \( E[\phi_{C}(A)] = 2 \cdot \phi_{C_{OPT}}(A) \), and (ii) \( \Pr[\phi_{C}(A) < 8\phi_{C_{OPT}}(A)] \geq \frac{3}{4} \).

Lemma 2. Let \( A \) be an arbitrary cluster in \( C_{OPT} \), and let \( C \) be an arbitrary clustering. If we add a random center to \( C \) from \( A \), chosen with \( D^2 \) weighting to get \( C' \), then (i) \( E[\phi_{C'}(A)] \leq 8\phi_{C_{OPT}}(A) \), and (ii) \( \Pr[\phi_{C}(A) < 8\phi_{C_{OPT}}(A)] \geq \frac{3}{4} \).

Let \( \mathcal{C} = \{A_1, \ldots, A_k\} \) denote the set of clusters in the optimal clustering \( C_{OPT} \). Let \( C^c \) denote the clustering after \( i \)th round of choosing centers. Let \( A^c \) denote the subset of clusters \( \mathcal{C} \) such that

\[
\forall A \in A^c, \phi_{C^c}(A) \leq 32 \cdot \phi_{C_{OPT}}(A).
\]

\(^2\)In many traditional online settings each part is a single data point.
Let this subset of clusters be called the ‘covered’ clusters. Let \( A_i^u = A_i \setminus A_i^c \) be the subset of ‘uncovered’ clusters. The following lemma shows that with constant probability step(1) of BTSC picks a center such that at least one of the clusters gets covered, i.e., \(|A_i^u| \geq 1\). Let us call this event \( E \). Then we have the following lemma.

**Lemma 3.** \( Pr[E] \geq (1 - \frac{1}{k}) \)

**Proof.** The proof follows from Lemma 2. ■

Let \( X_i^c = \bigcup_{A_i \in A_i^c} A_i \) and \( X_i^u = X - X_i^c \). Now, after the \( i^\text{th} \) round, either \( \phi_{C_i}(X_i^c) \leq \phi_{C_i}(X_i^u) \) or otherwise. In the former case, using Lemma 2, we show that the probability of covering an uncovered cluster in the \((i+1)^\text{th}\) round is large. In the latter case, we will show that the current set of centers is already competitive with constant approximation ratio. Let us start with the latter case.

**Lemma 4.** If event \( E \) occurs, i.e., \(|A_i^u| \geq 1\) and for any \( i > 1 \), \( \phi_{C_i}(X_i^c) > \phi_{C_i}(X_i^u) \), then \( \phi_{C_i} \leq 64\phi_{OPT} \).

**Proof.** We get this result using the following sequence of inequalities:
\[
\phi_{C_i} = \phi_{C_i}(X_i^c) + \phi_{C_i}(X_i^u) \leq \phi_{C_i}(X_i^c) \leq 2 \cdot 32 \cdot \phi_{OPT}(X_i^c) \leq 64\phi_{OPT}.
\]

**Lemma 5.** If for any \( i \geq 1 \), \( \phi_{C_i}(X_i^c) \leq \phi_{C_i}(X_i^u) \), then \( Pr[|A_i^u| \geq |A_i^u| + 1] \geq (1 - \frac{1}{k}) \).

**Proof.** We note that in the \((i+1)^\text{th}\) round, the probability that a center is chosen from a cluster \( \notin A_i^c \) is at least \( \frac{\phi_{C_i}(X_i^c)}{\phi_{C_i}(X_i^c) + \phi_{C_i}(X_i^u)} \geq \frac{1}{2} \).
Conditioned on this event, with probability at least \( \frac{3}{4} \) any of the centers \( x \) chosen in round \((i+1)\) satisfies \( \phi_{C_i \cup \{x\}}(A) \leq 32 \cdot \phi_{OPT}\). This further implies that with probability at least \( (1 - \frac{1}{k}) \) at least one of the chosen centers \( x \) in round \((i+1)\) satisfies \( \phi_{C_i \cup \{x\}}(A) \leq 32 \cdot \phi_{OPT}\). We now use Lemmas 4 and 5 to prove the main result of this section.

**Theorem 1.** BTSC is a \((O(\log k), O(1))\)-approximation algorithm.

**Proof.** From Lemma 3, we know that event \( E \), i.e., \(|A_i^u| \geq 1\) occurs. Given this, suppose for any \( i > 1 \), after the \( i^\text{th} \) round \( \phi_{C_i}(X_i^c) > \phi_{C_i}(X_i^u) \). Then from Lemma 4 we have \( \phi_{C_i} \leq 64\phi_{OPT} \). If no such \( i \) exists, then from Lemma 5 we get that the probability that there exists a cluster \( A \in A_i^c \) such that \( A \) is not covered even after \( k \) rounds is at most: \( (1 - (1 - \frac{1}{k})^k) \leq \frac{1}{2} \). So with probability at least \( \frac{1}{2} \), the algorithm covers all the clusters in \( A \). In this case from Lemma 5, we have \( \phi_C = \phi_{OPT} \leq 32\phi_{OPT} \). Thus, we have shown that BTSC is a randomized algorithm for clustering which with probability at least \( \frac{1}{2} \) gives a clustering with competitive ratio 64. ■

### C. Optimal Number of Clusters

It is some sense a norm to pre-assume the number of clusters before running any time-series clustering algorithm [26], simply because computing the optimal number of clusters for a batch data set is a difficult problem. In this section we state a heuristic method to find the optimal number of clusters given a batch time series data set. The motivation behind stating this method is as follows: we have already proven via Theorem 1 that \( O(k \log k) \) number of clusters are enough to guarantee an optimal clustering for a given input \( k \). Thus, between \( k \) and \( k \log k \), there exists an integer (both boundaries inclusive) that is the theoretical optimum value for the number of clusters for a given data set. Using proper heuristics one could pin down this optimal value within a good accuracy level. Our stated heuristic method is based on the Silhouette Index in [4] and estimates the cluster cohesion (within or intra-variance) and the cluster separation (between or inter-variance) and combines them to compute a quality measure. Extensive experimental results conducted in [4] indicate that Silhouette Index is the best performing index. This fact motivates us to recommend this index for finding the optimal number of clusters for a given time series data set.

The Silhouette Index for a given clustering configuration \( C \) is defined as follows:
\[
Sil(C) = \frac{1}{N} \sum_{c_k \in C} \sum_{x_i \in c_k} \frac{b(x_i, c_k) - a(x_i, c_k)}{\max\{a(x_i, c_k), b(x_i, c_k)\}},
\]
where
\[
a(x_i, c_k) = \frac{1}{|c_k|} \sum_{x_j \in c_k} D(x_i, x_j),
\]
and
\[
b(x_i, c_k) = \min_{c_l \in \{C - c_k\}} \left\{ \frac{1}{|c_l|} \sum_{x_j \in c_l} D(x_i, x_j) \right\}.
\]

For a given time series data set, by fixing different values of \( k \) ranging in the interval \([k, k \log k]\), we can compare each clustering configuration for respective values of \( k \), and choose the \( k \) that gives the best \( Sil \) score.
V. ONLINE TIME SERIES CLUSTERING

In the previous section, we devised a batch clustering algorithm that takes as input, \( k \), the number of desired clusters, and outputs a clustering \( C \) with \( ak \) clusters such that \( \frac{\phi_C}{\phi_{COPT}} \leq b = O(1) \) in the worst case. In this section, using our batch clustering algorithm, we propose our online clustering algorithm. The first part of this section describes the online algorithm in detail, and the second part analyses the algorithm. We conclude this section by describing a variant of our proposed time series clustering algorithm.

A. The Algorithm

We provide our online time series clustering algorithm, OTSC, the principle behind which is a simple batch divide-and-conquer scheme, analyzed by [20] with respect to the \( k \)-medoid objective, and we use it here to approximate the \( k \)-means objective in an online setting. In Section 4, we designed BTSC, a variant of the batch \( k \)-means algorithm to cluster a given data set competitively into a configuration \( C \) having \( ak \) clusters such that \( \frac{\phi_C}{\phi_{COPT}} \leq b = O(1) \). OTSC uses BTSC in small batches to process chunks of incoming data points in an intermediate fashion, and once the entire data set has been scanned, combines results from the intermediate processing to output the final clustering configuration. The main rationale behind the divide and conquer scheme is to combine efficient batch algorithms on small data sets to form an efficient online algorithm for the entire data set.

Algorithm 2: OTSC finds a time series clustering

Input: (a) Point Set \( S \subset \mathbb{R}^d \) of consumer time series data. Let \( n = |S| \), (b) Number of desired clusters, \( k \epsilon \mathbb{N} \), (c) \( A \), a BTSC type \((a, b)\)-approximation algorithm to the \( k \)-means objective, (d) \( A' \), a BTSC type \((a', b')\)-approximation algorithm to the \( k \)-means objective.

Output: A clustering configuration, \( C \)

1. Divide \( S \) into groups \( S_1, S_2, ..., S_l \)
2. for \( i \leftarrow 1 \) to \( l \) do
3. \( \text{Run } A \text{ on } S_i \text{ to get } \leq ak \text{ centers } T_i = \{t_{i1}, t_{i2}, ...\} \)
4. Denote the induced clusters of \( S_i \) as \( S_{i1} \cup S_{i2} \cup .... \)
5. \( S_w \leftarrow T_1 \cup T_2 \cup .... \cup T_l \), with weights \( w(t_{ij}) \leftarrow |S_{ij}| \)
6. Run \( A' \) on \( S_w \) to get \( \leq a'k \) centers \( T \)
7. return \( T \)

Here, for \( A \) we use the \((3 \log k, 64)\) randomized approximation algorithm, \( k \)-means\#, in [1]. For \( A' \) we use the \((1, O(\log k))\) randomized approximation algorithm, \( k \)-means++, also in [1]. The basic idea behind OTSC is that for a huge data stream, we read as much of it as will fit into the memory (call this portion \( S_1 \)), solve this sub-instance, then read the next batch \( S_2 \), solve this sub-instance, and so on. At the end the partial solutions need to be combined to get the full solution. A figurative explanation

Fig. 1: Conceptualizing the Divide and Conquer Approach
is given in Figure 1. Note here that when every group $S_i$ has size $\sqrt{n/k}$, the algorithm takes just one pass, and takes $O(\sqrt{n/k})$ memory.

B. Algorithm Analysis

We have the following result related to the performance guarantee provided by algorithm OTSC.

**Theorem 2.** For any two $(a, b)$ - randomized approximation algorithms $A$ and $A'$, Algorithm OTSC outputs a clustering that is an $(a', 2b + 4b'(b + 1))$ approximation to the $k$-means objective.

**Proof.** Let $T^* = \{t^*_1, \ldots, t^*_k\}$ be the optimal $k$-means for data set $S$. Let $t^*(x) \in T^*$ be the mean closest to point $x$. Likewise, let $t(x) \in T$ be the point in $T$ closest to $x$, and $t_i(x)$ be the point in $T_i$ closest to $x$. Since we are dividing the data into subsets $S_i$, we will need to talk about the costs of clustering these subsets as well as the overall cost of clustering $S$. We define by $cost(S', T')$ to be the cost of means $T'$ for data $S'$. Thus, $cost(S', T')$ equals $\sum_{x \in S'} D(x, T')$ when the data points are unweighted, and equals $\sum_{x \in S'} w(x) D(x, T')$, when the points are weighted. Here, $D(x, T')$ is the distance of $x$ to the closest point in set $T'$. The $a'$ part in the theorem statement is obvious. The rest of the theorem statement will be proved over the following lemmas.

**Lemma 6.** $cost(S, T) \leq 2 \sum_{i=1}^l cost(S_i, T_i) + 2 \cdot cost(S_w, T)$.

**Proof.** Recalling the fact that $S_w$ consists of all means $t_{ij}$, with weights $w(t_{ij}) = |S_{ij}|$, we have

$$cost(S, T) \leq \sum_{i=1}^l \sum_{x \in S_i} D(x, T)^2 \leq \sum_{i=1}^l \sum_{x \in S_i} (D(x, t_i(x)) + D(t_i(x), T))^2,$$

or

$$cost(S, T) \leq 2 \sum_{i=1}^l \sum_{x \in S_i} D(x, t_i(x))^2 + 2 \sum_{i=1}^l \sum_{x \in S_i} D(t_i(x), T)^2,$$

or

$$cost(S, T) \leq 2 \sum_{i=1}^l cost(S_i, T_i) + 2 \sum_{i=1}^l \sum_{j} |S_{ij}| D(t_{ij}, T)^2,$$

or

$$cost(S, T) \leq 2 \sum_{i=1}^l cost(S_i, T_i) + 2 \cdot cost(S_w, T).$$

Hence, we have proved Lemma 6. □

The next lemma says that when clustering a data set $S'$, picking centers from $S'$ is at most twice as bad as picking centers from the entire underlying metric space, $\chi$.

**Lemma 7.** For any $S' \subset \chi$, we have

$$\min_{T' \subset S'} cost(S', T') \leq 2 \cdot \min_{T' \subset \chi} cost(S', T').$$

**Proof.** Let $T' \subset \chi$ be the optimal solution chosen from $\chi$. For each induced cluster of $S'$, replace its center $t' \in T'$ by the closest neighbor of $t'$ in $S'$. This at most quadruples the cost, by first, the application of the triangle inequality, and then followed by the fact that $(\alpha + \beta)^2 \leq \alpha^2 + \beta^2$ (due to the squared distance concept in $k$-means algorithm.). Hence we have proved Lemma 7. □

Our final goal is to upper-bound $cost(S, T)$, and we will do so by bounding the two terms on the right-hand side of Lemma 1. Let us start with the first of them. We'd certainly hope that $\sum_i cost(S_i, T_i)$ is smaller than $cost(S, T^*)$; after all, the former uses way more representatives (about $akt$ of them) to approximate the same set $S$. We now give a coarse upper bound to this effect.

**Lemma 8.**

$$\sum_{i=1}^l cost(S_i, T_i) \leq b \cdot cost(S, T^*).$$

**Proof.** Each $T_i$ is a $b$-approximation solution to the $k$-means problem for $S_i$. Thus

$$\sum_{i=1}^l cost(S_i, T_i) \leq \sum_{i=1}^l b \cdot \min_{T' \subset \chi} cost(S_i, T'),$$
or from Lemma 7 we have,
\[ \sum_{i=1}^{l} \text{cost}(S_i, T_i) \leq \sum_{i=1}^{l} b \cdot \min_{T' \subseteq \chi} \text{cost}(S_i, T'), \]
or
\[ \sum_{i=1}^{l} \text{cost}(S_i, T_i) \leq \sum_{i=1}^{l} b \cdot \text{cost}(S_i, T^*) = b \cdot \text{cost}(S, T^*). \]

Hence we have proved Lemma 8. ■

We now bound the second term on the right-hand side of Lemma 6.

**Lemma 9.**

\[ \text{cost}(S_w, T) \leq 2b'(\sum_{i=1}^{l} \text{cost}(S_i, T_i) + \text{cost}(S, T^*)). \]

**Proof.** It is enough to upper bound \( \text{cost}(S_w, T^*) \) and then invoke the following
\[ \text{cost}(S_w, T) \leq b' \cdot \min_{T' \subseteq S_w} \text{cost}(S_w, T'), \]
and
\[ b' \cdot \min_{T' \subseteq S_w} \text{cost}(S_w, T') \leq b' \cdot \min_{T' \subseteq \chi} \text{cost}(S_w, T'), \]
and
\[ b' \cdot \min_{T' \subseteq \chi} \text{cost}(S_w, T') \leq b' \cdot \text{cost}(S_w, T^*). \]

To do so we just need to apply the triangle inequality. We have
\[ \text{cost}(S_w, T^*)^2 = \sum_{ij} |S_{ij}| D(t_{ij}, T^*)^2, \]
where
\[ \sum_{ij} |S_{ij}| D(t_{ij}, T^*)^2 \leq 2 \sum_{i,j} (D(x, t_{ij}) + D(x, t^*(x)))^2, \]
or
\[ \sum_{ij} |S_{ij}| D(t_{ij}, T^*)^2 \leq 2 \sum_{i,j} D(x, t_{ij})^2 + 2 \sum_{i,j} D(x, t^*(x))^2, \]
or
\[ \text{cost}(S_w, T)^2 \leq 2 \sum_{i} \sum_{x \in S_{ij}} (D(x, t_i(x))^2) + 2 \sum_{i} \sum_{x \in S_{ij}} D(x, t^*(x))^2, \]
or
\[ \text{cost}(S_w, T)^2 \leq 2 \sum_{i} \text{cost}(S_i, T_i) + \text{cost}(S, T^*). \]

The proof of Theorem 2 follows immediately by putting together Lemma 6, 8, and 9. ■

We now proceed to analyzing the time and space complexity of our algorithm. We note from algorithm OTSC that \( A \) is the \( k \)-means \# algorithm that runs on the data \( 3 \log n \) times independently, and picks the clustering with the smallest cost. \( A' \) is just a single run of the \( k \)-means++ algorithm.

**Algorithmic Time and Space Analysis:** From the analysis and results in Section 4 in relation to algorithm BTSC, we have with probability at least \( 1 - (\frac{3}{2})^{3 \log n} \geq 1 - \frac{1}{n} \), algorithm A is a \( (3 \log k, 64) \)-approximation algorithm. Moreover, the space requirement remains logarithmic in the input size. In step (3) of algorithm OTSC, we run \( A \) on batches of data. Since each batch is of size \( \sqrt{n}k \), the number of batches is \( \sqrt{\frac{n}{k}} \), the probability that \( A \) is a \( (3 \log k, 64) \)-approximation algorithm for all the batches is at least \( (1 - \frac{1}{n})^{\sqrt{\frac{n}{k}}} \geq \frac{1}{2} \). Conditioned on this event, the divide and conquer strategy gives a \( O(\log k) \)-approximation algorithm. The memory required is \( O(\log k \cdot \sqrt{n}k) \) times the logarithm of the input size. Moreover, the algorithm has a running time of \( O(dnk \log n \log k) \).
C. A Variant to OTSC

In a particular variant of OTSC, we could incrementally output a $k$ clustering configuration after processing each batch of data, i.e., after seeing batch $i$, using intermediate computations for batches $1$ to $i-1$, we could have a $k$-clustering configuration. Since $(1 - \frac{1}{p})^p \geq \frac{1}{2}$ for any $p$ in the range $[1, \sqrt{n}]$, the optimality guarantees for this variant of OTSC would be the same as that for OTSC. The running time of this variant is $O(ldnk \log n \log k)$ for $l$ batches of data. We have the following result related to Algorithm, V-OTSC.

**Theorem 3.** For any two $(a, b)$ - randomized approximation algorithms $A$ and $A'$, Algorithm V-OTSC outputs a clustering in $O(ldn \log n \log k)$ time, that is an $(a', 2b + 4b(b + 1))$ approximation to the $k$-means objective.

**Proof.** The proof is very similar to that of Theorem 2. ■

We now state the algorithm.

**Algorithm 3:** V-OTSC finds a time series clustering

**Input:** (a) Point Set $S \subset \mathbb{R}^d$ of consumer time series data. Let $n = |S|$, (b) Number of desired clusters, $k \in \mathbb{N}$, (c) $A$, a $BTS$ type $(a, b)$-approximation algorithm to the $k$-means objective, (d) $A'$, a $BTS$ type $(a', b')$-approximation algorithm to the $k$-means objective.

**Output:** A clustering configuration, $C$

1. Divide $S$ into groups $S_1, S_2, \ldots, S_l$
2. for $i \leftarrow 1$ to $l$
   3. for $j \leftarrow 1$ to $i$
   4. Run $A$ on $S_j$ to get $\leq ak$ centers $T_j = \{t_{j1}, t_{j2}, \ldots\}$
   5. Denote the induced clusters of $S_j$ as $S_{j1} \cup S_{j2} \cup \ldots$
   6. $S_w \leftarrow T_1 \cup T_2 \cup \ldots \cup T_i$, with weights $w(t_{ij}) \leftarrow |S_{ij}|
   7. Run $A'$ on $S_w$ to get $\leq a'k$ centers $T$
6. return $T$

VI. EXPERIMENTAL EVALUATION

While the theoretical analysis of the proposed online algorithm has shown that in the worst case scenario it performs better than the existing heuristics approaches. However this does not say anything about its average behavior for real-life data sets.

Hence, we empirically evaluate the performance of our online time series clustering approach (see Section V) in practice for scalable prediction of energy consumption in Smart Grids. It is well known that while the consumption of a single consumer is hard to forecast [40], that of a large group of users can be accurately forecast [24, 25]. Aggregation can dramatically reduce the overall uncertainty in customers’ consumption by splitting customers into groups, which aggregate consumption is represented by virtual customers [24, 25]. When forming the aggregates, customers can be grouped arbitrarily [39], based on geographical information [25, 16, 33], or based on some formal cost function [39, 25]. For example, PG&E divides Northern California into several zones [25]. As there might be significant behavior variation, even within in a single zip code, it is important to optimize the clustering process to obtain the "best" results.

Here, we propose the use of our online clustering technique (see Section V) to group customers together into virtual customers, with the goal of minimizing the cumulative consumption prediction error. Time series forecasting models such as ARIMA [40] can be trained on historical consumption data aggregated from multiple customers to predict the near future based on real-time data transmitted by smart meters. We use ARIMA for prediction as it has been shown to outperform other time series forecasting models for short term prediction of electricity consumption [40]. The goal of our incremental clustering approach is to computationally scale the problem of grouping customers whose cumulative energy consumption behaves in a more predictable way and minimizes the ARIMA prediction error for the cluster as a whole.

Our numerical simulations are based on synthetic times series data generated according to the process described in [2]. Specifically, we produce representative data for 50 smart meters over a three month period at 15-minute intervals. We note that each interval is a dimension. Thus, each data point (individual user time series of power consumption) is approximately of $3 \times 30 \times 96 = 8,600$ dimensions. We used the first two months data per smart meter as a training set for the standard Auto-Regressive Integrated Moving Average (ARIMA) method used for time series predictive analysis, and the last month’s data as an evaluating set for calculating the Mean Absolute Percentage Error (MAPE) of the ARIMA model. We show results for 3 sets of synthetic data of 50 smart meters each. Our experiments were run using the MATLAB package. We empirically analyze clustering performance and prediction accuracy. Specifically, (i) we compare the clustering performance (clustering cost) of our proposed algorithm with a traditional batch $k$ means method, and (ii) we compare the average MAPE obtained from our
clustering method with that obtained by the method proposed in [39]. Our work, while validated for this vital CPS application domain, is applicable to scalable clustering of other large-scale time series data sets.

In the worst case, our algorithm will always perform (refer to Section 5.2, Thm 2.) better than the worst case behavior of the heuristic. However, the theory does not comment on non worst-case behavior, and worst cases might rarely arise unless an intelligent adversary (refer to Section 7.) is able to enforce it pretty often. One of our main motivations behind empirical evaluation is to gauge the extent of predictive performance improvement of our proposed online time-series clustering algorithm over recent heuristic approaches [39] that performed well over general large non-synthetic data sets, and under non worst case scenarios. The rationale here is that a better performance of our algorithm over a heuristic that performed well for real data sets would be sort of an indication of our algorithm performing better than proposed heuristics for real-data sets, irrespective of initial seeding configuration.

Our results in Figure 2 show that the clustering performance of the OTSC algorithm is better than the traditional batch \( k \)-means algorithm in most cases, where performance is measured by the cost incurred by a clustering algorithm. Please note that the TSC algorithm in the plots refer to the online clustering algorithm, i.e., the OTSC algorithm in our paper. In fact, with increasing number of clusters, OTSC increasingly outperforms the batch \( k \)-means algorithm. Via Figure 3 we show that the average MAPE is lower with the application of our proposed OTSC algorithm than with the application of the incremental algorithm proposed in [39]. In addition, there is a linear increase of average MAPE with the increase in the number of clusters with TSC compared to the non-linear increasing pattern observed in [39]. The reason for the increase in the MAPE with increasing number of clusters is identical to the one mentioned in [39], i.e., with less number of consumers per cluster the average MAPE is more than in the case when there are more consumers per cluster and less clusters.

VII. DISCUSSION

In this section we discuss two important topics related to our clustering work: (a) the role of an adversary in the clustering process, and (b) the impact of parallel and distributed architectures on online clustering.

A. The Role of an Adversary

An Added Motivation: In Section 1, we stated the importance of quality guarantees with respect to clustering and motivated our work on the counts of bad initial seeding possible with the clustering approaches of existing works (refer to Section 2.).
However, it is also quite possible for a malicious adversary to hack the mechanism of existing segmentation algorithms that results in an unboundedly bad segmentation. Previous research efforts based on clustering heuristics fail to address adversarial issues and this is a major drawback, given the increase risks of cyber-physical systems to adversarial attacks. To put this fact more clearly, existing solutions work well under trustworthy environments, but looking forward to the future they might not stand up to potent adversaries. Thus, an added motivation to design clustering methods applied to cyber-physical applications that can theoretically guarantee worst case performance in the presence of adversaries.

**The Way Adversaries Work:** In a recent series of works [7][8], the authors educate us on ways adversaries can manipulate data (in our problem, time series data) before they enter the clustering engine of an utility. The main idea in these works is that an attacker may significantly poison the whole clustering process by adding a relatively small percentage of attack samples to the input data, and that some attack samples may be obfuscated to be hidden within some existing clusters without excessively altering the initial clustering. The analysis in the papers have focused on the single-and complete-linkage hierarchical clustering algorithms, but the attack methods extend to other clustering algorithms as well.

**B. The Role of Parallel and Distributed Computing**

While the focus of this paper is to propose a polynomial time online clustering algorithm for limited memory systems it can be argued that existing parallel and distributed systems such as clouds do not face this constraint and can be used as a viable alternative with any existing parallel clustering algorithm. The rationale lies in the fact that the processing of the time series batches \( S_t \) in Figure 1 can be easily done in parallel using state of the art techniques such as Streaming Map Reduce [9, 43] for data streams. In addition the possible bottleneck caused by aggregating the results \( S_n \) could be solved by using elastic solutions that scale out depending on the system’s load [10]. Our online algorithm can easily be integrated into such a system providing an efficient means of processing data stream chunks in parallel faster than existing heuristics in a worst case scenario.

**VIII. Conclusion and Future Work**

In this paper, we proposed an efficient online time series clustering approach to perform predictive analysis of energy consumption of consumers in Smart Grid for the purposes of effective demand response. Compared to previous approaches, our clustering method provides provable guarantees on the optimality of clustering, and is applicable to Big Data pertaining to large metropolitan localities. In lieu of the fact that an adversary can maliciously manipulate the functioning of existing customer segmentation algorithms to enable poor clustering configurations, providing performance guarantees on the optimality of clustering is an important problem. We adopted tools from the theory of randomized approximation algorithms to design our clustering method, which also accounts for high dimensional, high volume data. We showed that for a given number of clusters it is possible to have a clustering configuration that is within \( O(\log k) \) of the optimal clustering configuration. Through initial experiments we show that our algorithm outperforms existing algorithms in literature in terms of contributing to reduction in MAPE, even for situations where existing algorithms might not be victim to an adversary or initial bad seeding. In addition, we also provided a mechanism to find that particular value of the number of clusters that gives the best approximate clustering configuration for a given times series data set.

In this paper we ran initial experiments using synthetically generated data sets to validate that our theory not only addresses worst case adversarial scenarios but also performs better than existing heuristics in terms of predictive analysis, under non-adversarial settings and situations when initial seeding for existing heuristics is not bad. As part of future work, we plan to extend our experimental setup to include real data from a utility company, and at the same time run more detailed experiments to characterize the impact of our algorithm on predictive performance. Given a time horizon, we provided a method to cluster data for a given number of consumers during that time horizon. An interesting extension would be to design a smart way to update the clusters for incoming data on future intervals, for a given fixed set of consumers. This case is much more frequently likely than the case of clustering new consumers for a given fixed time horizon. We plan to address this issue as part of future work.

**Acknowledgement**

This material is based upon work supported by the United States Department of Energy under Award Number number DE-OE0000192, and the Los Angeles Department of Water and Power (LA DWP). The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, the LA DWP, nor any of their employees.

**References**


