Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com

ELSEVIER

A parametric bootstrap algorithm for cluster number determination of load pattern categorization

Xing Luo [a, b], Xu Zhu [a, c, *], Eng Gee Lim [b]

[a] The Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, L69 3GJ, UK
[b] The Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou, 215123, China
[c] The School of Electronic and Information Engineering, Harbin Institute of Technology, Shenzhen, 518055, China

ARTICLE INFO

Article history:
Received 17 December 2018
Received in revised form 26 March 2019
Accepted 15 April 2019
Available online xxx

Keywords:
Load pattern categorization
Parametric bootstrap algorithm
Cluster number determination
Cascade clustering

ABSTRACT

The latest development of smart grid technologies gives rise to big load data and requires load pattern categorization (LPC). How to determine a precise cluster number and choose an appropriate clustering algorithm are critical and still remain challenging in LPC. In this work, we propose a novel parametric bootstrap (PB) algorithm to address the cluster number determination problem in load pattern analysis. The proposed PB algorithm is more robust against dimensionality of data and more applicable for the load demand data which is usually of high dimensionality. The PB algorithm is also general and independent of data type, resulting in a more precise cluster number determined than existing methods with little fluctuation. Moreover, an effective cascade clustering scheme is proposed to categorize load demand data and analyze load patterns, based on the PB algorithm and the K-means++ clustering algorithm. The results indicate the feasibility and the superiority of the proposed approach.

1. Introduction

Load pattern categorization (LPC) refers to the process of clustering similar electricity consumption patterns into clusters. It is an established yet active research topic due to its widespread applications in smart grid. The recent ongoing development of smart grid technologies for data acquisition and supervision, metering, and communication, also gives rise to huge volume of load data which can offer vast benefits with respect to LPC research [1]. The potential applications of LPC can be summarized in four major aspects.

(1) Power system planning and operation. In smart grid, the electricity suppliers are operating with a competitive environment as the electricity distribution and supply services have been unbundled. The electricity suppliers need to get accurate information on the actual load demand of their users for setting up dedicated commercial offers, thus improving the planning and operation of power system [2,3]. Customer grouping on the basis of similar load demand pattern is likely to provide an effective solution.
(2) Demand response. Enhanced knowledge on LPC can be decidedly useful to support demand response (DR) in smart grid. LPC has been proposed as effective means for enhancing targeting and tailoring of DR programs as well as providing reasonable load scheduling recommendations, owning to availability of advanced technology for load shifting and to emerging opportunities for flexible demand management, producing incentives and rewards to the participating users [4,5].
(3) Load demand forecasting. LPC plays a crucial role in load demand forecasting (LDF) in smart grid, which is an essential part of power generation, distribution and regulation. LDF usually relies on available data from similar days and it is often estimated by the aggregation of typical load patterns (TLPs) which are the outcomes of LPC [6–8]. Obviously, an effective and precise LPC can provide relevant information so as to improve the performance of LDF.
(4) Electricity tariff formulating. LPC is also proposed for the purpose of setting variable electricity tariffs in smart grid. Electricity suppliers now have some degrees of freedom in formulating tariff offers which can meet the requirements set by regulatory authorities. However, each tariff is formulated with reference to a specific load category, defined by a number of load characteristics [9]. Additionally, LPC also can be used to assist electricity consumers in adequately selecting an appropriate tariff [10] in an electricity market.

As having extensive applications in the industrial field, a wide variety of clustering technologies have been conducted and applied to load demand data. There are many ways for clustering technique classification. According to different clustering objectives, the clustering technologies can be generally summarized as three categories: partition-based methods, hierarchical methods and model-based methods. The partition-based methods include K-means [11,12], K-medoids [13,14] and other generations of K-means (*e.g.,* fuzzy K-means [15] and Kernel K-means [16]). In addition, the main repre-

[*] Corresponding author. The Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, L69 3GJ, UK.
Email address: xuzhu@liv.ac.uk (X. Zhu)

https://doi.org/10.1016/j.energy.2019.04.089
0360-5442/ © 2019.

sentatives of the hierarchical methods are agglomerative clustering and divisive clustering [17]. An alternative approach, beyond partition-based and hierarchical clustering methods, is the use of distribution mixture models. Gaussian mixture models [18,19] and C/D-vine Copula mixture models [20,21]) are outstanding candidates in model-based methods. Besides, the approaches of using wavelet transforms [22,23], neural network [24,25] and other machine learning algorithms for data clustering become increasingly popular in recent years. Several clustering algorithms can be applied to a certain dataset, however, there is no single method best for all datasets. Hence, we determine a more effective clustering method which is appropriate for load data clustering instead of developing a new method in this work.

Meanwhile, a common problem in the most existing clustering methods is that the number of clusters (called *k*) has been assumed to be a pre-defined parameter, which is difficult to set in practice. It is not always clear what is the best value for *k*. Nevertheless, using an imprecise cluster number as the input in LPC significantly reduces the clustering accuracy of load data and increases the complexity [26]. Therefore, a reliable cluster number has to be determined ahead of clustering in LPC.

A number of algorithms including X-means [27,28], G-means [22,29], and other methods which determine the cluster number by finding an "inflection point" by certain criteria [20,30,31], have been proposed in the literature to determine the value of *k* automatically, mainly on the basis of K-means or similar clustering techniques.

Specifically, a regularization framework for learning the value of *k*, which is called X-means, was first proposed in Ref. [32]. The algorithm searches over many potential *k* values in the range [$k_{\min}$,$k_{\max}$] . With X-means, each cluster is treated as a parent cluster, which can be split into two children clusters according to the score of Bayesian information criterion (BIC) (Akaikes information criterion (AIC), is also acceptable in the usage of X-means). The scores help to determine whether the parent cluster or the children clusters are a better representative for the data. For example, the X-means algorithm was adopted in cluster number determination for load profile clustering of smart metering data in Ref. [27]. However, this algorithm is slow as it needs to rerun K-means for each cluster splitting.

In addition, the Gaussian-means (G-means) algorithm [22,29] also provides a way to determine an appropriate cluster number. G-means starts with a small number of K-means centers, and grows the number of centers. Each iteration of the algorithm splits into two centers whose data appear not to come from a Gaussian distribution via the Anderson-Darling (AD) statistic, which is a powerful 1-dimension test. The splitting continues until the data in all clusters pass the AD test so that the expected cluster number can be obtained. For example, the authors of [22] proposed a load pattern clustering strategy based on wavelet transformation and using G-means to determine the cluster number. In this way, the adopted load data of N-dimension has to be reduced to a single dimension, as the G-means algorithm is not effective for highly dimensional data. However, the actual load data of a typical day is usually in 24-dimension or 48-dimension, and the dimensional-reduction always gives rise to the risk of information loss. Therefore, the G-means algorithm is not proper for the data that is of multi-dimension.

Another popular approach of cluster number determination is to find an inflection point by certain criteria, such as the AIC based method or BIC based method [20,30,31]. For example, a mixture model for residential load data clustering was presented in Ref. [20]. Authors selected the optimal cluster number by seeking the first knee always at the local maximal of the curve of AIC. However, the AIC based algorithm is not reliable and cannot guarantee to find a precise cluster number, since the estimated inflection point normally varies in a range in AIC calculations.

Motivated by the above open issues, in this paper, we propose a novel parametric bootstrap (PB) algorithm to address the cluster number determination problem in LPC and incorporate it with compatible clustering techniques. The main contributions of our work are summarized as follows.

(1) The proposed PB algorithm is more robust against dimensionality of the data in LPC than conventional methods (*e.g.*, G-means [22]). In particular, it can effectively determine the cluster number for the data in high dimensional space, for which the previous methods [22,29] are not applicable. Therefore, the proposed PB algorithm is applicable to analyzing load demand data, which is usually of 24-dimension or 48-dimension.
(2) The proposed PB algorithm is general and independent of data type. It is more reliable and stable in cluster number determination than the existing methods (*e.g.*, G-means [22] and the AIC based algorithm [20]), with much higher probability of successfully finding a precise cluster number and lower standard deviation (STD) value.
(3) An effective cascade clustering scheme which classifies the initial load data into a series of sub-cascades according to external features, is proposed to reduce the clustering errors and improve the efficiency over clustering the raw data directly. Besides, the proposed PB algorithm is incorporated with various clustering techniques [11–14,18,19], among which K-means++ demonstrates the best performance in LPC.

The rest of this paper is organized as follows. A cascade clustering scheme is presented specifically in Section 2. In Section 3, the parametric bootstrap algorithm with compatible clustering techniques is illustrated in details. The feasibility and reliability of the proposed approach are also evaluated. Afterwards, the verified approach is applied to the actual load data to address the cluster number determination problem and obtain the objective TLPs in Section 4. The clustering performances are compared. Finally, the paper is concluded concisely in Section 5.

2. Cascade clustering scheme for load data preprocessing

In this section, we propose a cascade clustering scheme that comprises two major stages for load data processing, as illustrated in Fig. 1.

Based on the observations of load demand in different time in UK (as shown in Fig. 2), the total load demand of electricity consumers is significantly influenced by external factors with apparent facts:

(1) The load demand of weekends is evidently less than that of weekdays, even though the trends are similar.
(2) The weekly periodicity of the load series is broken by the occurrence of a UK bank holiday, as shown in Fig. 2(b).
(3) The load shapes of UK bank holidays are also dissimilar to both weekdays and weekends.
(4) The load demand apparently varies with seasonality, as shown in Fig. 2(a) and (c).

According to these, pre-clustering the initial load demand data into a series of sub-cascades at the first stage of the proposed cascade clustering scheme is significant, and obviously it is capable of reducing the clustering errors and improving the efficiency compared with clustering the raw data directly.

As a result, the initial load demand data is divided into $i \times j$ sub-cascades, where *i* and *j* are referred to as the day type and the

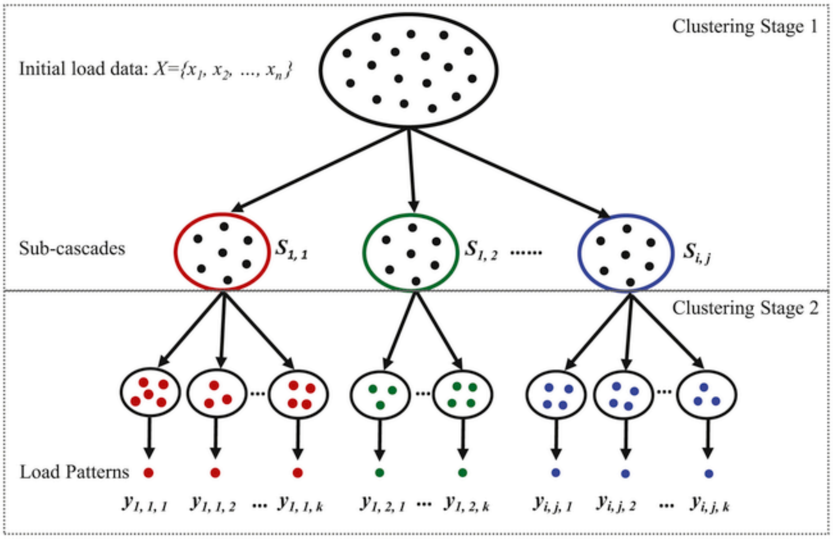
Clustering Stage 1
Initial load data: $X=\{x_1, x_2, \ldots, x_n\}$
Sub-cascades
$S_{1,1}$
$S_{1,2}$ ......
$S_{i,j}$
Clustering Stage 2
Load Patterns
$y_{1,1,1}$ $y_{1,1,2}$ ... $y_{1,1,k}$
$y_{1,2,1}$ ... $y_{1,2,k}$
$y_{i,j,1}$ $y_{i,j,2}$ ... $y_{i,j,k}$

Fig. 1. Cascade clustering scheme for load data processing.

Stage 1: A set of load demand data is classified into a series of sub-cascades based on the external features including the seasonality and the day type.
Stage 2: Objects in each individual sub-cascade are further categorized into numbers of clusters based on the PB algorithm incorporated with a compatible clustering technique.

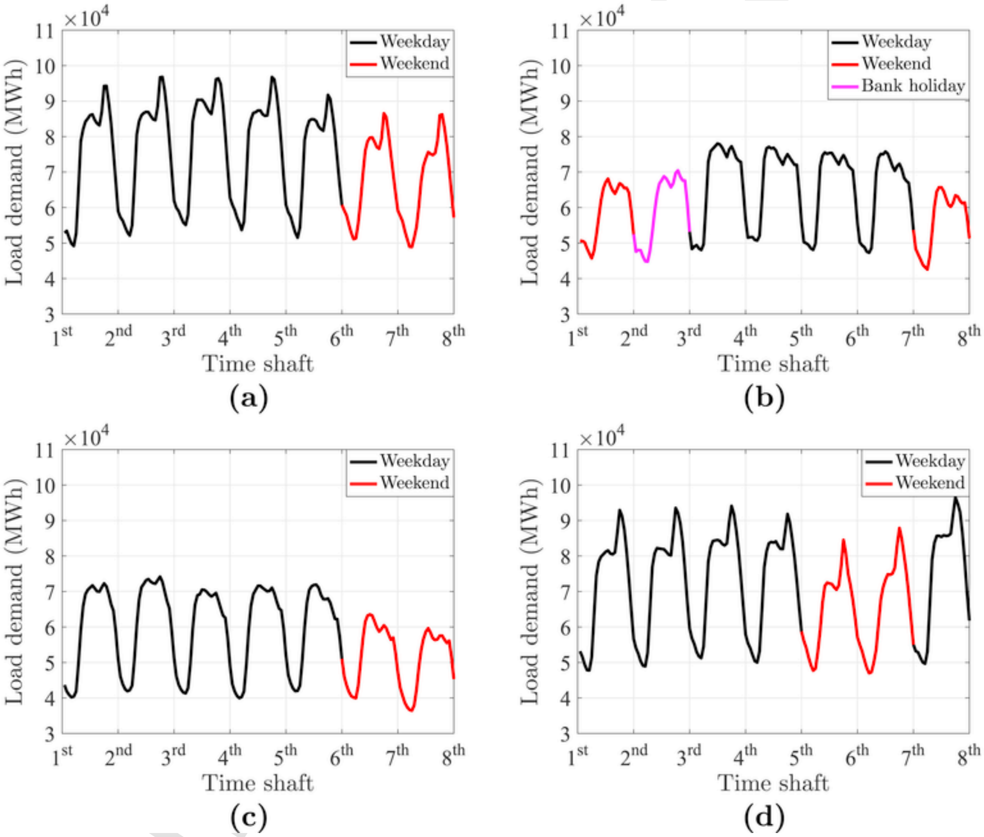

Fig. 2. Load demand observations of different time in a year. (a) Data of February; (b) Data of May; (c) Data of August; (d) Data of November.

seasonality, respectively. In this work, the day types are considered as "working day" consisting of weekdays excluding UK bank holidays, "non-working day" consisting of weekends excluding UK bank holidays and "UK bank holiday", *i.e.,* Good Friday, Easter Monday, Christmas Day, *etc*. Meanwhile, the seasonality is divided by month.

Following this, the second stage of the cascade scheme focuses on finding the internal relationships between objects within the same sub-cascade and allocate the objects into refined clusters, ensuring that the objects within the same cluster are similar. In terms of sub-cascade clustering, a precise cluster number *k*, has to be determined at first. Thus, we propose a robust parametric bootstrap (PB) algorithm to resolve the cluster number determination problem and it is incorporated with a compatible clustering technique to cluster load data simultaneously. Afterwards, the typical load patterns (TLPs) of each sub-cascade can be extracted as $y_{i,j,k}$, as shown in Fig. 1. Note that in a practical implementation, the sub-cascade clustering can be executed in parallel, thus making this a potentially very fast scheme.

3. Parametric bootstrap algorithm for precise cluster number determination

This section illustrates the parametric bootstrap (PB) algorithm incorporated with a compatible clustering technique to determine a precise cluster number in details. The proposed PB algorithm is more robust against dimensionality of the data in LPC than conventional methods and has a better performance of successfully finding a precise cluster number with little fluctuation. It is verified in subsection 3.3.

3.1. Parametric bootstrap algorithm

The PB algorithm can be considered as a bootstrap re-sampling based technique, which estimates the number of components by incrementally testing the hypothesis that there are $k + 1$ components against the null hypothesis that there are *k* components via parametric bootstrap. An accepted *k* value is determined based on the significance level (SL) of the hypotheses. The whole process of determine a precise cluster number by the proposed PB algorithm is briefly summarized in Algorithm 1.

Specifically, given an input dataset of a random sub-cascade $\mathcal{S}$ with *N* objects, it can be mathematically characterized as:

$$\mathcal{S} = \left\{x_n, n = 1, 2, \ldots, N, x_n \in \mathbb{R}^d\right\};$$

where *d* denotes the dimensionality which indicates the resolution of a load curve. To begin with, the input data is firstly hypothesized as consisting of *k* clusters and it is categorized into *k* groups by using a compatible clustering technique.

In addition, the feature parameter vector of a cluster, $P = \{\mu, c, E\}$, consisting of mean vector μ, covariance *c* and a $d \times d$ covariance matrix *E*, can be obtained according to Equations (1)–(3), respectively.

Mean vector μ:

$$\mu = \frac{\sum_{n=1}^{N_k} x_n}{N_k} \tag{1}$$

Covariance *c*:

$$c\left(x^{(a)}, x^{(b)}\right) = \frac{\sum_{n=1}^{N_k}\left(x_n^{(a)} - \mu^{(a)}\right)\left(x_n^{(b)} - \mu^{(b)}\right)}{N_k - 1} \tag{2}$$

Covariance matrix *E*:

$$E = \begin{bmatrix} c\left(x^{(1)}, x^{(1)}\right) & c\left(x^{(1)}, x^{(2)}\right) & \cdots & c\left(x^{(1)}, x^{(d)}\right) \\ c\left(x^{(2)}, x^{(1)}\right) & c\left(x^{(2)}, x^{(2)}\right) & \cdots & c\left(x^{(2)}, x^{(d)}\right) \\ \vdots & \vdots & \ddots & \vdots \\ c\left(x^{(d)}, x^{(1)}\right) & c\left(x^{(d)}, x^{(2)}\right) & \cdots & c\left(x^{(d)}, x^{(d)}\right) \end{bmatrix} \tag{3}$$

where N_k is the total number of objects within k^{th} cluster among *K* clusters and $a, b \in [1, d]$. The feature parameter *P* is obtained from the given dataset and it is used to generate synthetic data. Afterwards, sets of synthetic data with the same size as the real data can be generated periodically based on the obtained feature parameters. The process of generating the synthetic data is the core of the algorithm and it is called "bootstrap simulation" (BS).

Moreover, the sum of square errors (SSE, denoted as Φ) which is the summation of the squared distance of each point within a cluster from the cluster center, is proposed to evaluate the clustering quality. Hence, SSE of i^{th} BS data set is demonstrated in Equation (4). The probability density function (PDF) over a number of Φ^{BS} (denoted as $F(\Phi^{\text{BS}})$) also can be obtained.

$$\Phi_i^{\text{BS}} = \sum_{k=1}^{K}\sum_{n=1}^{N_k}\sum_{d=1}^{D}\left(x_{k,n}^{\text{BS},(d)} - \mu_k^{\text{BS},(d)}\right)^2 \tag{4}$$

Further, to assess the hypothesis that a dataset is composed of $k + 1$ clusters against the null hypothesis that it has only *k* clusters, the actual data is clustered into $k + 1$ clusters and SSE of the actual dataset on the hypothesis *k* can be obtained in Equation (5).

$$\Phi^{\text{AC}} = \sum_{k=1}^{K+1}\sum_{n=1}^{N_k}\sum_{d=1}^{D}\left(x_{k,n}^{(d)} - \mu_k^{(d)}\right)^2 \tag{5}$$

Algorithm 1 Parametric bootstrap algorithm for cluster number determination

```
Input: dataset $\mathcal{S} = \{x_n, n = 1, 2, \ldots, N, x_n \in \mathbb{R}^d\}$.
Output: cluster number $k$.
1: hypothesize an initial $k$.
2: classify $\mathcal{S}$ into $k$ clusters.
3: obtain the feature parameter vector $P = \{\mu, c, E\}$
4: generate sets of BS data utilizing $P$ vector.
5: calculate $\Phi_i^{\text{BS}}$ of each BS dataset.
6: generate PDF over a number of $\Phi_i^{\text{BS}}$.
7: classify $\mathcal{S}$ into $k + 1$ clusters, calculate related $\Phi^{\text{AC}}$.
8: calculate p-value of the hypothesis $k$.
9: if p-value satisfies the requirement: $p \geq \alpha$
10: then accept the hypothesis $k$ as an appropriate cluster number.
11: else $k \leftarrow k + 1$
12: repeat step 2 - step 10.
```

Although the real dataset is actually consisting of *k* clusters, the performance of $k + 1$ clusters (or more) is normally "better" than simulations, since the data is categorized into smaller and tighter clusters. Thus, the objective is to find the rate at which SSE decreases will slow down for *k* beyond the objective cluster number. Accordingly, the *p* value (proposed in Equation (6)) which is a widely used parameter in a statistical hypothesis [33] is adopted to determine a precise

k.

$$p = \int_{-\infty}^{\Phi^{\mathrm{AC}}} F\left(\Phi^{\mathrm{BS}}\right) \cdot d\Phi^{\mathrm{BS}} \tag{6}$$

In the proposed PB algorithm, the requirement of an acceptable hypothesis k is defined as: if $p < \alpha$, the hypothesis of including k clusters is rejected and we tend to hypothesize that the data has at least $k + 1$ clusters. The evaluating process continues with the increase of the values of k until it satisfies the condition $p \geq \alpha$, where α is an accepted threshold in statistical hypothesis and is generally set to 0.01 or 0.05 [34].

3.2. Compatible clustering technique: K-Means++

In order to execute the PB algorithm, a compatible clustering technique is required. A number of clustering algorithms can be used to assist the proposed PB algorithm to determine the cluster number as well as clustering data. However, there is no single algorithm best for all datasets. In this work, K-means++ which is an improved version of K-means algorithm is selected as the classifier due to its higher efficiency and improved robustness compared with others (*e.g.,* standard K-means, K-medoids, Gaussian mixture models, *etc.*).

A good clustering result satisfies the condition that the distance between arbitrary two clusters should be as far as possible [35]. Intuitively, it is wise to choose the initial centers that are far away from each other in the beginning in the K-means++ algorithm. Except for the first center that is chosen uniformly and randomly from the data samples, each subsequent center is chosen from the remaining data samples with the probability proportional to its squared distance from the determined cluster center that is closest to the point. This special seeding can greatly increase the convergence speed of the algorithm.

After having initial centers, for a dataset $\mathcal{S} = \left\{x_n, n = 1, 2, \ldots, N, x_n \in \mathbb{R}^d\right\}$, the algorithm divides $\mathcal{S}$ into k exhaustive clusters $\Omega = \left\{\Omega_k, k = 1, 2, \ldots, K\right\}$, $\cup_{i=1}^{k}\Omega_i = \mathcal{S}$, $\Omega_i \cap \Omega_j = \emptyset$ for $1 \leq i \neq j \leq K$. For a cluster, the center is given by:

$$\omega_i = \frac{1}{\left|\Omega_i\right|}\sum_{x \in \Omega_i} x \tag{7}$$

where Ω_i is the center of i^{th} cluster. Let $\omega = \left\{\omega_1, \ldots, \omega_k\right\}$ be a set of centers and $\left\|x_i - x_j\right\|$ represent the Euclidean distance between x_i and x_j. The objective of K-means++ is to find an optimal ω to minimize:

$$\arg\min_{\Omega} \sum_{k=1}^{K} \sum_{x \in \Omega_k} \left\|x - \omega_k\right\|_2 \tag{8}$$

Algorithm 2 K-means++

```
Input: cluster number $k$, dataset $\mathcal{S} = \left\{x_n, n = 1, 2, \ldots, N, x_n \in \mathbb{R}^d\right\}$.
Output: centers $\omega = \left\{\omega_1, \omega_2, \ldots, \omega_k\right\}$.
1: $\omega \leftarrow \emptyset$.
2: Initialization Step:
3: Choose one center $x$ from $\mathcal{S}$ at random, $\omega = \omega \cup x$.
4: Choose $x \in \mathcal{S}$ with probability: $\frac{D(x)^2}{\sum_{x \in \mathcal{S}} D(x)^2}$, $\omega = \omega \cup x$.
5: Repeat step 4 until $k$ centers are chosen.
6: Assignment Step:
7: Assign each object $x_p$ to a cluster $\Omega^{(t)}$ according to:
   $\Omega_i^{(t)} = \left\{x_p : \left\|x_p - \omega_i^{(t)}\right\|_2 \leq \left\|x_p - \omega_j^{(t)}\right\|_2 \forall j, 1 \leq j \leq k\right\}$.
8: Update Step:
9: Update centers of clusters according to:
   $\omega_i^{(t+1)} = \frac{1}{\left|\Omega_i^{(t)}\right|}\sum_{x_j \in \Omega_i^{(t)}} x_j$.
10: Repeat step 6 - step 9 until $\omega$ becomes stable.
```

The K-means++ algorithm tries to put object x into a cluster Ω_k to be similar to each other whilst being dissimilar to objects in other clusters, which is similar to the standard K-means algorithm. It takes the cluster number as the input parameter and k initial centers are specifically selected. Afterwards, the remaining objects are assigned to the clusters with the closest centers according to the similarity. The algorithm continues to update the means of clusters until the means converge and become stable. Let $D(x)$ be the Euclidean distance between x and the nearest center that has already been chosen. Hence, the K-means++ algorithm that proceeds by alternating three steps, the initialization step, the assignment step and the update step can be illustrated concisely in Algorithm 3.2.

Compared with the standard K-means, K-means++ guarantees to find a solution that is $O(logk)$ competitive to the optimal K-means solution, which means that K-means++ has better effectiveness in clustering than the standard K-means algorithm. Although the special seeding in K-means++ takes extra time, its clustering part converges fast so that the clustering efficiency is significantly improved. The K-means++ algorithm also demonstrates a better clustering performance in LPC compared with other clustering techniques such as K-medoids and GMM. It is presented in subsection 4.5.

3.3. Algorithm verification

It is impossible to evaluate the feasibility of algorithms on the actual load data, since the data is not labeled by groups. Therefore, this section proposes unsupervised examples to assess the effectiveness of the proposed PB algorithm. The tested dataset actually consists of 4 random Gaussian components in 24-dimensional space, which is similar to the actual load demand data. The hypothesis of k normally starts with a small number (*e.g.,* $k = 2$) and increases gradually until a satisfied k is obtained. In this case, the hypotheses of $k = 2$ to 5 are presented. For each hypothesis, $N^{\mathrm{BS}} = 5 \times 10^5$ bootstrap simulations are generated. The amount of BS data is usually set according to the requirements that an accurate PDF of Φ^{BS} can be obtained. Additionally, the significance thresholds $\alpha_1 = 0.01$ and $\alpha_2 = 0.05$ are both accounted in the evaluation. The proposed PB algorithm can be proved valid and effective while the evaluation result corresponds with the setup, *i.e.,* the obtained cluster number is equal to 4.

Fig. 3 presents the cluster number determination result of the proposed testing data based on the PB algorithm incorporated with the K-means++ clustering technique, where the red curves denote SSE of the actual data (Φ^{AC}) in $k + 1$ clustering and the black curves represent PDF of total square errors of BS data (Φ^{BS}). Specifically, the result shows that the p-values of the cases $k = 2$ and $k = 3$ in Fig. 3(a) and (b), respectively, are equal to 0, which indicates that the hypotheses are rejected and the dataset includes at least $k + 1$ clusters. In addition, when $k = 4$ in Fig. 3(c), the p-value increases to 0.197, which is greater than the pre-defined significance threshold α. Therefore, the hypothesis $k = 4$ is accepted as a precise cluster number according to the cluster number determination requirement in subsection 3.1. Moreover, it can be seen that the p-value (0.334) of hypothesis $k = 5$ in Fig. 3(d) is greater than the p-value (0.197) of hypothesis $k = 4$ as well as greater than α. The result conforms to the

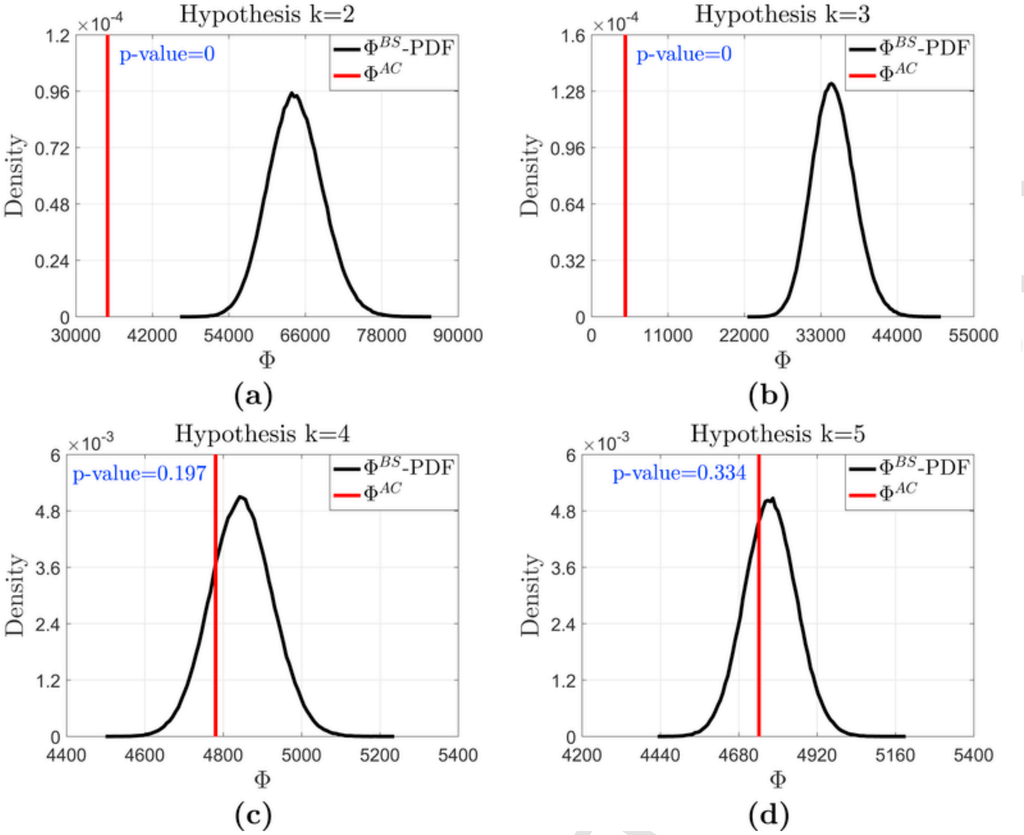

Fig. 3. Cluster number determination for a 24-dimensional space dataset based on the PB algorithm incorporated with K-means++. Hypotheses of $k=2$ to 5 are evaluated.

expectation that the rate of Φ decreases with the k value increasing. Therefore, the hypothesis of $k = 4$ is regarded as a precise cluster number, which also corresponds to the initial setup in this case.

Compared with the most popular methods, G-means [22] and the AIC based algorithm [20], which also can be used in the cluster number selection, the PB algorithm is more robust and reliable, particularly in processing the high dimensional space data. Specifically, the G-means algorithm is not effective for highly dimensional data such as the load data, since it cannot ensure the data within all dimensions simultaneously pass the Anderson-Darling (AD) statistic which is a powerful 1-dimension test. One solution is to reduce the data of N-dimension to a single dimension [22]. However, the dimensional-reduction always gives rise to the risk of information loss and failures of obtaining a cluster number always occur.

On the other hand, the AIC based algorithm is also compared. The AIC based algorithm determines the precise cluster number by seeking the first knee always at the local maximal of the curve of AIC. However, the AIC based algorithm also cannot guarantee to find a precise cluster number as the estimated inflection point normally varies in a range in AIC calculations.

In order to evaluate the reliability of an algorithm, a performance metric of failure rate (FR) is defined as:

$$FR = \frac{N_{\text{failure}}}{N_{\text{test}}} \tag{9}$$

where N_{failure} and N_{test} represent the numbers of failed tests and total tests, respectively.

The comparison results of algorithms in terms of the probability of finding an actual cluster number, the standard deviation (STD) and FR, over $N_{\text{test}} = 10^2$ tests are presented in Table 1.

The results show that the proposed PB algorithm is more effective in the cluster number determination than G-means and AIC based algorithms, with much higher probability (0.97) of successfully finding an actual cluster number. In addition, the PB algorithm is more robust against dimensionality of the data than G-means ($\text{FR}_{\text{G-means}} = 0.28$). However, the dimensional conditions have few effects ($\text{FR}_{\text{PB}} = 0$) upon the outcome of cluster number selection by using the PB algorithm. Moreover, the PB algorithm is also more stable than the AIC based algorithm with a lower STD value ($\text{STD}_{\text{PB}} = 0.17$) (as a number of tests by G-means failed to obtain a cluster number, the G-means algorithm is not accounted in STD evaluation). In summary, although G-means and AIC based algorithms can achieve the correct cluster number in some cases, both algorithms cannot guarantee to find a precise cluster number at all times due to their inherent defects.

Table 1
Performance comparison of cluster number determination between algorithms.

Criterion	Algorithm		
	PB	G-means	AIC based algorithm
Probability	0.97	0.45	0.28
STD	0.17	–	1.03
FR	0	0.28	0

4. Case study

The verified PB algorithm for cluster number determination incorporated with the K-means++ clustering algorithm is applied to real-life load data in this section. The initial load dataset is classified into a number of sub-cascades at first according to the proposed scheme described in Section 2. Afterwards, the cluster numbers of selected sub-cascades are determined and typical load patterns (TLPs) are derived.

4.1. Case descriptions

In this study, a large set of historical load demand data on national level provided by the National Grid Ltd, UK [36] is adopted. The dataset includes 3653 days' load demand data with time intervals 2 h ($d = 12$) and 0.5 h ($d = 48$) from the year 2007–2016.

As objects in the initial dataset are classified into a series of sub-cascades, we simply utilize mathematical numbers to label the seasonal features and the day types of each sub-cascade. Specifically, the numbers "1–12" are used to label 12 months in different seasons and "1–3" are used to indicate the day types ("1"→ "working day", "2" → "none-working day" and "3" → "UK bank holiday"). For instance, $\mathscr{S}_{5,1}$ denotes the sub-cascade of working day in May. The rest sub-cascades can be deduced by analogy. Additionally, due to limited data samples on special events such as UK bank holidays that can be collected (8 UK bank holidays per year and 82 UK bank holidays in total), the exclusive sub-cascade $\mathscr{S}_{\forall,3}$ consists of all load information of UK bank holidays. Other types of special events data can be analyzed in a similar way.

Due to the limited space, 9 typical sub-cascades (*i.e.,* $\mathscr{S}_{2,1}$, $\mathscr{S}_{5,1}$, $\mathscr{S}_{8,1}$, $\mathscr{S}_{11,1}$, $\mathscr{S}_{2,2}$, $\mathscr{S}_{5,2}$, $\mathscr{S}_{8,2}$, $\mathscr{S}_{11,2}$ and $\mathscr{S}_{\forall,3}$) covering various scenarios of load data are selected as examples to perform the results.

4.2. Evaluation metrics

In order to evaluate the similarity of TLPs between a variety of sub-cascades, the Pearson correlation coefficient (PCC, denoted as ρ) which is a measure of the linear correlation between two variables X and Y in statistics, is proposed in Equation (10).

$$\rho(X,Y) = \frac{\sum_{d=1}^{D}\left(x_d - \bar{x}\right)\cdot\left(y_d - \bar{y}\right)}{\sqrt{\sum_{d=1}^{D}\left(x_d - \bar{x}\right)^2}\cdot\sqrt{\sum_{d=1}^{D}\left(y_d - \bar{y}\right)^2}} \tag{10}$$

where D is the sample size of the compared time series TLPs. x_d, y_d are the individual sample points indexed with d. $\bar{x} = \sum_{d=1}^{D} x_d / D$ and analogously for $\bar{y}$. PCC has a value between +1 and − 1, where +1 indicates the total positive linear correlation, 0 represents no linear correlation, and − 1 denotes the total negative linear correlation between X and Y.

In terms of assessing the clustering performance between a variety of algorithms, a number of evaluation metrics [6,22] are introduced. The proposed metrics which are object distance based, mainly evaluate the compactness performance between objects within the same cluster and the separation performance between disparate clusters.

Given a sub-cascade $\mathscr{S}$, we assume the objects in $\mathscr{S}$ are classified into K clusters. For an individual cluster Ω_i, $x_{i,n}$ and ω_i represent an object within Ω_i and the center of Ω_i, respectively. Hence, the evaluation metrics including compactness (CP), separation (SP) and Davies-Bouldin index (DBI) can be presented in Equations (11)–(14), respectively, as follows.

Compactness (CP) metric:

$$\mathrm{CP} = \frac{1}{K}\sum_{i=1}^{K}\overline{\mathrm{CP}}_i \tag{11}$$

$$\overline{\mathrm{CP}}_i = \frac{1}{\left|\Omega_i\right|}\sum_{x_{i,n}\in\Omega_i}\left\|x_{i,n} - \omega_i\right\| \tag{12}$$

where $\overline{\mathrm{CP}}_i$ is the averaged taxicab distance between objects $x_{i,n}$ and the cluster center ω_i within the cluster Ω_i. Metric of CP shows the compactness or homogeneity between objects within clusters. The smaller value of CP indicates the more compact of objects within a cluster.

Separation (SP) metric:

$$\mathrm{SP} = \frac{2}{K^2 - K}\sum_{i=1}^{K}\sum_{j=i+1}^{K}\left\|\omega_i - \omega_j\right\|_2 \tag{13}$$

Metric of SP describes the separation or the distance between clusters. The larger value of *SP* illustrates the greater distance between the centroid of clusters.

Davies-Bouldin index (DBI) metric:

$$\mathrm{DBI} = \frac{1}{K}\sum_{i=1}^{K}\max_{j\neq i}\left(\frac{\overline{\mathrm{CP}}_i + \overline{\mathrm{CP}}_j}{\left\|\omega_i - \omega_j\right\|_2}\right) \tag{14}$$

Metric of DBI represents the system-wide average of the similarity measures of each cluster with its most similar cluster. The lower value of DBI, the better performance.

In addition, the sum of square errors (Φ, proposed in subsection 3.1) which is the most significant metric, is also considered in the evaluation of accurate clustering. Moreover, as the time consuming of algorithms executed on disparate platforms is different, the relative running time (RRT) which is defined in Equation (15), is taken to evaluate the efficiency between algorithms.

$$\mathrm{RRT} = \frac{\mathrm{CT}_i}{\mathrm{CT}_{\min}} \tag{15}$$

where CT_i denotes the running time of the i^{th} algorithm and $\mathrm{CT}_{\min}$ is the minimal running time among all the algorithms of comparison.

4.3. Cluster number determination of load demand data

One necessary and significant task before clustering objects into several clusters is to determine an appropriate cluster number. In this subsection, the cluster number determination program for each sub-cascade has been run for multiple times, to ensure a reliable result. The significance levels (SL) $\alpha_1 = 0.01$ and $\alpha_2 = 0.05$ are both taken into account. Based on the verified PB algorithm incorporated with K-means++, the cluster number determination results for the load data of 9 typical sub-cascades are presented in Table 2.

Table 2
Results of the cluster number determination with p-values for the typical sub-cascades using the PB algorithm incorporated with K-means++.

Dimension	SL	Sub-cascade								
		$\mathcal{S}_{2,1}$	$\mathcal{S}_{5,1}$	$\mathcal{S}_{8,1}$	$\mathcal{S}_{11,1}$	$\mathcal{S}_{2,2}$	$\mathcal{S}_{5,2}$	$\mathcal{S}_{8,2}$	$\mathcal{S}_{11,2}$	$\mathcal{S}_{\forall,3}$
d = 12	$\alpha_1 = 0.01$	k = 4 p = 0.017	k = 4 p = 0.040	k = 4 p = 0.039	k = 4 p = 0.036	k = 3 p = 0.057	k = 3 p = 0.038	k = 3 p = 0.027	k = 3 p = 0.042	k = 3 p = 0.110
	$\alpha_1 = 0.05$	k = 5 p = 0.064	k = 6 p = 0.073	k = 5 p = 0.066	k = 5 p = 0.053	k = 3 p = 0.057	k = 4 p = 0.051	k = 4 p = 0.060	k = 4 p = 0.064	k = 3 p = 0.110
d = 48	$\alpha_1 = 0.01$	k = 4 p = 0.017	k = 4 p = 0.030	k = 4 p = 0.012	k = 4 p = 0.019	k = 3 p = 0.037	k = 3 p = 0.037	k = 3 p = 0.018	k = 3 p = 0.049	k = 3 p = 0.133
	$\alpha_1 = 0.05$	k = 5 p = 0.068	k = 5 p = 0.086	k = 5 p = 0.064	k = 5 p = 0.120	k = 4 p = 0.115	k = 4 p = 0.070	k = 4 p = 0.105	k = 4 p = 0.090	k = 3 p = 0.133

Specifically, the results illustrate that the cluster numbers of the "working day" in sub-cascades $\mathcal{S}_{2,1}$, $\mathcal{S}_{5,1}$, $\mathcal{S}_{8,1}$ and $\mathcal{S}_{11,1}$ are normally equal to 4 for $\alpha_1 = 0.01$, and 5 for $\alpha_2 = 0.05$. However, the cluster numbers of the "none-working day" in sub-cascades $\mathcal{S}_{2,2}$, $\mathcal{S}_{5,2}$, $\mathcal{S}_{8,2}$ and $\mathcal{S}_{11,2}$ are usually equal to 3 for $\alpha_1 = 0.01$, and 4 for $\alpha_2 = 0.05$. In addition to these, the exclusive sub-cascade $\mathcal{S}_{\forall,3}$ is suggested to be classified into 3 clusters. The obtained cluster numbers are used as input parameters to ensure the data classification and TPLs extraction in LPC.

4.4. Load pattern categorization

In order to reduce the risk of ending up in a local optimum, the clustering process has been executed for multiple time with random initializations. Besides, the obtained cluster number which is under the significant level $\alpha_1 = 0.01$, is adopted as an input parameter in the categorization. The center of a cluster which leads to a minimal SSE of the cluster, is regarded as one of the TLPs within a sub-cascade.

While the final TLPs are regarded as the aggregation of TLPs within each individual sub-cascade under the cascade clustering scheme.

The obtained TLPs of 9 typical sub-cascades covering a variety of scenarios are proposed as examples in Fig. 4. Specifically, the light grey curves represent the actual load demand data that is required to be categorized and the colorized curves are the objective TLPs. The results show that the extracted TLPs follow the shape of actual load curves and cover the most actual load curves within the same sub-cascade. The proportion of objects categorized into different TLPs is presented in Table 3.

In addition, Table 4 illustrates the evaluation of similarity between TLPs of relevant sub-cascades based on the metric of PCC. High correlations between TLPs are marked in bold. It can be seen that the achieved coefficients of the compared TLPs are very high (≥ 0.837) in general, which means that there are positive relationships among TLPs. Moreover, the coefficients between TLPs within the same sub-cascades are extremely high (≥ 0.97). On the contrary, the coefficients between TLPs from different sub-cascades are relatively low for most cases. The facts indicate the higher correlations between TLPs within the same sub-cascade and relatively weaker relationships between TLPs from the different sub-cascades. It is also in line with the analysis in Section 2.

4.5. Clustering performance evaluation

The clustering performance is another issue we may concern. The performance comparison of different clustering techniques incorporated with the PB algorithm for $d = 12$ and $d = 48$ are presented in Tables 5 and 6, respectively. The best performance metrics are marked in bold. Obviously, the K-means++ algorithm outperforms in most metrics in overall comparison. Specifically, K-means++ performs the best in Φ and CP evaluations, which means the objects clustered by K-means++ algorithm are more compact and have less square errors. In addition, in terms of SP and DBI evaluations, K-means++ also performs the best in some cases, such as $\mathcal{S}_{5,1}$, $\mathcal{S}_{8,1}$, $\mathcal{S}_{11,1}$ and $\mathcal{S}_{2,2}$. On the contrary, the GMM algorithm has the worst clustering performance in the evaluation and it is not effective in clustering the data in high dimensional space, such as $d = 48$.

Moreover, the adopted K-means++ algorithm is also the most efficient clustering algorithm as it spends the minimal running time among all compared algorithms. In accordance with the RRT results in Tables 5 and 6, K-means++ is 2.11, 3.93 and 381 times faster than K-means, K-medoids and GMM, respectively in average.

Much more interesting, we notice that the clustering performance of the K-means algorithm is similar to the K-means++ algorithm, except the RRT evaluation. This is because the K-means++ algorithm is improved based on the standard K-means algorithm. The difference is that the special seeding in K-means++ significantly accelerates the convergence process in K-means. Besides, the multiple simulations which are taken in our work significantly avoid the risk of ending up in a local optimum for the K-means++ algorithm.

5. Discussion and conclusion

This paper proposed an innovative parametric bootstrap algorithm incorporated with K-means++ clustering technique, to address the cluster number determination problem as well as clustering the load data simultaneously in LPC. A number of evaluations were presented in this work, which indicated that our approach is more robust and reliable in finding a precise cluster number than conventional methods and it also has a better performance in load pattern clustering compared with literature methods.

In this paper, we mainly focus on addressing cluster number determination problem in clustering, rather than developing a new cluster algorithm. In fact, for a given dataset, the data clustering performance is largely influenced by the choice of clustering algorithm, the input cluster number and the characteristics of the data. Several clustering algorithms can be applied to a certain dataset, however, there is no single clustering method that is best for all datasets. Therefore, we have utilized a relatively robust and efficient clustering technique, K-means++, instead of developing a new method in this work.

As the proposed PB algorithm is entirely unsupervised, it also can be widely adopted in solving the cluster number determination problem for a variety of data processing tasks. The future research can focus on testing the PB algorithm with other suitable clustering techniques and apply the algorithms to other datasets in industry.

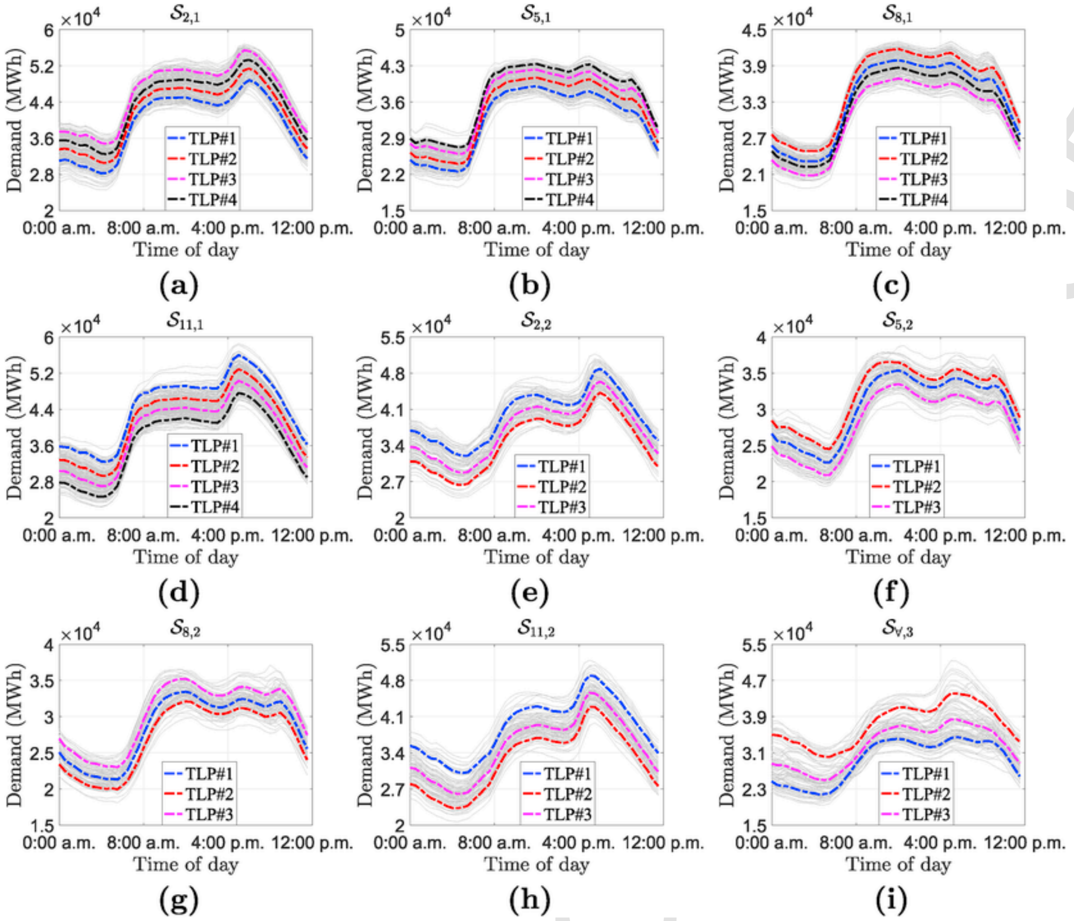

Fig. 4. Extracted TLPs of 9 selected sub-cascades. The objective TLPs are colorized. (a) Sub-cascade $\mathcal{S}_{2,1}$; (b) Sub-cascade $\mathcal{S}_{5,1}$; (c) Sub-cascade $\mathcal{S}_{8,1}$; (d) Sub-cascade $\mathcal{S}_{11,1}$; (e) Sub-cascade $\mathcal{S}_{2,2}$; (f) Sub-cascade $\mathcal{S}_{5,2}$; (g) Sub-cascade $\mathcal{S}_{8,2}$; (h) Sub-cascade $\mathcal{S}_{11,2}$; (i) Sub-cascade.$\mathcal{S}_{\forall,3}$

Table 3
Proportion of objects categorized into different TLPs.

TLP	Sub-cascade								
	$\mathcal{S}_{2,1}$	$\mathcal{S}_{5,1}$	$\mathcal{S}_{8,1}$	$\mathcal{S}_{11,1}$	$\mathcal{S}_{2,2}$	$\mathcal{S}_{5,2}$	$\mathcal{S}_{8,2}$	$\mathcal{S}_{11,2}$	$\mathcal{S}_{\forall,3}$
TLP #1	24.2%	23.9%	36.2%	10.8%	35.0%	42.2%	45.6%	18.4%	41.5%
TLP #2	28.6%	36.3%	20.5%	30.1%	22.5%	23.3%	27.8%	21.8%	19.5%
TLP #3	17.2%	29.4%	13.8%	39.9%	42.5%	34.5%	26.6%	59.8%	39.0%
TLP #4	30.0%	10.4%	29.5%	19.2%	–	–	–	–	–

Table 4
Similarity evaluation between TLPs of 3 typical sub-cascades based on the metric of PCC.

TLPs		$\mathcal{S}_{2,1}$				$\mathcal{S}_{8,2}$			$\mathcal{S}_{\forall,3}$		
		TLP#1	TLP#2	TLP#3	TLP#4	TLP#1	TLP#2	TLP#3	TLP#1	TLP#2	TLP#3
$\mathcal{S}_{2,1}$	TLP#1	**1.000**	**0.999**	**0.998**	**0.998**	0.911	0.908	0.908	0.934	0.837	0.910
	TLP#2	**0.999**	**1.000**	**0.999**	**0.999**	0.910	0.908	0.907	0.934	0.848	0.915
	TLP#3	**0.998**	**0.999**	**1.000**	**0.999**	0.906	0.905	0.902	0.931	0.848	0.914
	TLP#4	**0.998**	**0.999**	**0.999**	**1.000**	0.915	0.914	0.911	0.939	0.858	0.923
$\mathcal{S}_{8,2}$	TLP#1	0.911	0.910	0.906	0.914	**1.000**	**0.998**	**0.999**	**0.994**	0.906	**0.973**
	TLP#2	0.908	0.908	0.905	0.914	**0.998**	**1.000**	**0.997**	**0.994**	0.919	**0.980**
	TLP#3	0.908	0.906	0.902	0.911	**0.999**	**0.997**	**1.000**	**0.993**	0.899	0.968
$\mathcal{S}_{\forall,3}$	TLP#1	0.934	0.934	0.931	0.939	**0.994**	**0.994**	**0.993**	**1.000**	0.923	**0.983**
	TLP#2	0.837	0.847	0.848	0.858	0.906	0.919	0.899	0.923	**1.000**	**0.976**
	TLP#3	0.910	0.915	0.913	0.922	**0.972**	**0.980**	0.968	**0.983**	**0.976**	**1.000**

Table 5
Performance comparison of different clustering techniques incorporated with the PB algorithm, *d* = 12.

Metric	Algorithms	Sub-cascade								
		$\mathcal{S}_{2,1}$	$\mathcal{S}_{5,1}$	$\mathcal{S}_{8,1}$	$\mathcal{S}_{11,1}$	$\mathcal{S}_{2,2}$	$\mathcal{S}_{5,2}$	$\mathcal{S}_{8,2}$	$\mathcal{S}_{11,2}$	$\mathcal{S}_{\forall,3}$
$\Phi^{\times 10^{10}}$	K-means++	**4.16**	**2.45**	**1.85**	**3.81**	**2.73**	**1.65**	**1.25**	**2.95**	**4.69**
	K-means	4.16	2.45	1.85	3.81	2.73	1.65	1.25	2.95	4.69
	K-medoids	4.61	2.76	1.96	4.12	3.29	1.93	1.42	3.31	5.58
	GMM	4.69	2.91	2.11	5.95	2.76	1.70	–	–	5.34
$\mathrm{CP}^{\times 10^{4}}$	K-means++	**3.91**	**2.93**	**2.64**	**3.72**	**5.10**	**3.68**	**3.15**	**5.13**	**6.61**
	K-means	3.91	2.93	2.64	3.72	5.10	3.68	3.15	5.13	6.61
	K-medoids	4.10	3.04	2.69	3.78	5.42	3.80	3.26	5.39	6.86
	GMM	4.15	3.22	2.78	4.39	5.15	3.75	–	–	7.92
$\mathrm{SP}^{\times 10^{4}}$	K-means++	4.88	**4.48**	**3.73**	**6.87**	**4.99**	3.38	3.18	6.62	7.45
	K-means	4.88	4.48	3.73	6.87	4.99	3.37	3.18	6.62	7.45
	K-medoids	**4.95**	3.60	3.64	5.96	4.97	**3.67**	**3.35**	**6.63**	**8.44**
	GMM	4.74	3.69	3.50	5.41	4.77	3.28	–	–	7.12
DBI	K-means++	2.80	**2.61**	2.62	**2.22**	**2.83**	3.10	2.69	2.54	2.46
	K-means	2.80	2.61	2.62	2.22	2.83	3.10	2.69	**2.51**	2.46
	K-medoids	**2.79**	3.11	**2.41**	2.40	2.97	**2.94**	**2.59**	2.55	**2.38**
	GMM	3.20	3.59	2.95	3.10	3.05	3.42	–	–	3.16
RRT	K-means++	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	K-means	2.29	2.01	2.13	2.16	2.79	1.74	2.50	2.99	3.03
	K-medoids	4.86	4.83	4.41	4.60	4.56	4.66	4.42	5.32	5.44
	GMM	583	581	446	703	120	113	–	–	122

Table 6
Performance comparison of different clustering techniques incorporated with the PB algorithm, *d* = 48.

Metric	Algorithms	Sub-cascade								
		$\mathcal{S}_{2,1}$	$\mathcal{S}_{5,1}$	$\mathcal{S}_{8,1}$	$\mathcal{S}_{11,1}$	$\mathcal{S}_{2,2}$	$\mathcal{S}_{5,2}$	$\mathcal{S}_{8,2}$	$\mathcal{S}_{11,2}$	$\mathcal{S}_{\forall,3}$
$\Phi^{\times 10^{10}}$	K-means++	**1.10**	**0.648**	**0.496**	**1.00**	**0.712**	**0.434**	**0.331**	**0.770**	**0.123**
	K-means	1.10	0.648	0.496	1.00	0.712	0.434	0.331	0.770	0.123
	K-medoids	1.26	0.739	0.536	1.09	0.863	0.513	0.383	0.879	0.149
	GMM	–	–	–	–	–	–	–	–	–
$\mathrm{CP}^{\times 10^{4}}$	K-means++	**4.01**	**3.00**	**2.70**	**3.80**	**5.19**	**3.73**	**3.24**	**5.23**	**6.75**
	K-means	4.02	3.00	2.70	3.80	5.19	3.74	3.24	5.23	6.75
	K-medoids	4.24	3.14	2.76	3.88	5.52	3.89	3.35	5.50	7.04
	GMM	–	–	–	–	–	–	–	–	–
$\mathrm{SP}^{\times 10^{4}}$	K-means++	2.44	**2.25**	**1.87**	**3.45**	**2.50**	1.70	1.59	3.31	3.73
	K-means	2.44	2.25	1.87	2.95	2.50	1.69	1.60	3.31	3.73
	K-medoids	**2.49**	1.82	1.81	2.98	2.42	**1.85**	**1.69**	**3.33**	**4.24**
	GMM	–	–	–	–	–	–	–	–	–
DBI	K-means++	5.75	**5.29**	**5.41**	**4.55**	**5.78**	6.31	5.51	5.16	5.06
	K-means	5.75	5.29	5.41	4.55	5.78	6.31	5.51	**5.12**	5.06
	K-medoids	**5.70**	5.93	5.44	5.01	6.04	**6.06**	**5.28**	5.18	**4.87**
	GMM	–	–	–	–	–	–	–	–	3.16
RRT	K-means++	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	K-means	1.48	1.34	1.45	1.31	1.95	2.25	2.10	2.27	2.23
	K-medoids	2.70	2.91	2.72	2.48	3.07	3.45	3.40	3.50	3.49
	GMM	–	–	–	–	–	–	–	–	122

Authors contributions

Xing Luo proposed this topic, carried out numerical experiments, and drafted the manuscript. Xu Zhu and Eng Gee Lim checked and clarified the manuscript carefully. All authors read and approved the final manuscript.

Ethics approval

This article does not contain any studies with human participants performed by any of the authors.

Consent for publication

Informed consent was obtained from all individual participants included in the study.

Conflicts of interest

The authors declare that they have no conflicts of interest.

Funding

This work is partially supported by the XJTLU Research Development Fund (PGRS-13-03-06, RDF-14-03-24 and RDF-14-02-48) and AI University Research Centre (AI-URC) through XJTLU Key Programme Special Fund (KSF-P-02).

Acknowledgements

This work is supported by the University of Liverpool and Xi'an Jiaotong-Liverpool University.

References

[1] G. Chicco, Overview and performance assessment of the clustering methods for electrical load pattern grouping, Energy 42 (1) (2012) 68–80, 8th World Energy System Conference, WESC 2010 https://doi.org/10.1016/j.energy.2011.12.031.

[2] T. Handa, A. Oda, T. Tachikawa, J. Ichimura, Y. Watanabe, H. Nishi, Knives: A distributed demand side management system-integration with zigbee wireless sensor network and application, In: Proc. 2008 6th IEEE International Conference on industrial Informatics, Daejeon, Korea, 2008, pp. 324–329, https://doi.org/10.1109/INDIN.2008.4618117.

[3] Y. Suhara, T. Nakabe, G. Mine, H. Nishi, Distributed demand side management system for home energy management, In: Proc. IECON 2010-36th Annual Conference on IEEE industrial Electronics Society, Glendale, USA, 2010, pp. 2430–2435, https://doi.org/10.1109/IECON.2010.5675426.

[4] H. Cao, C. Beckel, T. Staake, Are domestic load profiles stable over time? an attempt to identify target households for demand side management campaigns, In: Proc. IECON 2013-39th Annual Conference of the, IEEE Industrial Electronics Society, Vienna, Austria, 2013, pp. 4733–4738, https://doi.org/10.1109/IECON.2013.6699900.

[5] J. Kwac, J. Flora, R. Rajagopal, Household energy consumption segmentation using hourly data, IEEE Transactions on Smart Grid 5 (1) (2014) 420–430, https://doi.org/10.1109/TSG.2013.2278477.

[6] M. Chaouch, Clustering based improvement of nonparametric functional time series forecasting: Application to intra-day household-level load curves, IEEE Transactions on Smart Grid 5 (1) (2014) 411–419, https://doi.org/10.1109/TSG.2013.2277171.

[7] K.H.S.V.S. Nunna, S. Doolla, Responsive end-user-based demand side management in multimicrogrid environment, IEEE Trans Ind Inf 10 (2) (2014) 1262–1272, https://doi.org/10.1109/TII.2014.2307761.

[8] F. McLoughlin, A. Duffy, M. Conlon, A clustering approach to domestic electricity load profile characterisation using smart metering data, Appl Energy 141 (Supplement C) (2015) 190–199 https://doi.org/10.1016/j.apenergy.2014.12.039.

[9] C.S. Chen, J.C. Hwang, C.W. Huang, Application of load survey systems to proper tariff design, IEEE Trans Power Syst 12 (4) (1997) 1746–1751, https://doi.org/10.1109/59.627886.

[10] D. Fischer, B. Stephen, A. Flunk, N. Kreifels, K.B. Lindberg, B. Wille-Haussmann, E.H. Owens, Modeling the effects of variable tariffs on domestic electric load profiles by use of occupant behavior submodels, IEEE Transactions on Smart Grid 8 (6) (2017) 2685–2693, https://doi.org/10.1109/TSG.2016.2544141.

[11] F.L. Quilumba, W.J. Lee, H. Huang, D.Y. Wang, R.L. Szabados, Using smart meter data to improve the accuracy of intraday load forecasting considering customer behavior similarities, IEEE Transactions on Smart Grid 6 (2) (2015) 911–918, https://doi.org/10.1109/TSG.2014.2364233.

[12] K. Xing, C. Hu, J. Yu, X. Cheng, F. Zhang, Mutual privacy preserving k-means clustering in social participatory sensing, IEEE Trans Ind Inf 13 (4) (2017) 2066–2076, https://doi.org/10.1109/TII.2017.2695487.

[13] F. McLoughlin, A. Duffy, M. Conlon, A clustering approach to domestic electricity load profile characterisation using smart metering data, Appl Energy 141 (Supplement C) (2015) 190–199 https://doi.org/10.1016/j.apenergy.2014.12.039.

[14] A. Onan, A k-medoids based clustering scheme with an application to document clustering, In: Proc. 2017 International Conference on Computer Science and Engineering (UBMK), Antalya, Turkey, 2017, pp. 354–359, https://doi.org/10.1109/UBMK.2017.8093409.

[15] J. Qin, W. Fu, H. Gao, W.X. Zheng, Distributed k-means algorithm and fuzzy c-means algorithm for sensor networks based on multiagent consensus theory, IEEE Trans Cybern 47 (3) (2017) 772–783, https://doi.org/10.1109/TCYB.2016.2526683.

[16] J. Santarcangelo, X.P. Zhang, Dynamic time-alignment k-means kernel clustering for time sequence clustering, In: Proc. 2015 IEEE International Conference on image processing (ICIP), Quebec, Canada, 2015, pp. 2532–2536, https://doi.org/10.1109/ICIP.2015.7351259.

[17] S. Zhou, Z. Xu, F. Liu, Method for determining the optimal number of clusters based on agglomerative hierarchical clustering, IEEE Trans Neural Netw Learn Syst 28 (12) (2017) 3007–3017, https://doi.org/10.1109/TNNLS.2016.2608001.

[18] A. Ganjavi, E. Christopher, C.M. Johnson, J. Clare, A study on probability of distribution loads based on expectation maximization algorithm, In: Proc. 2017 IEEE power Energy Society innovative smart grid technologies Conference (ISGT), Washington D.C., USA, 2017, pp. 1–5, https://doi.org/10.1109/ISGT.2017.8086037.

[19] J. Zhang, Z. Yin, R. Wang, Pattern classification of instantaneous cognitive task-load through gmm clustering, laplacian eigenmap, and ensemble svms, IEEE ACM Trans Comput Biol Bioinform 14 (4) (2017) 947–965, https://doi.org/10.1109/TCBB.2016.2561927.

[20] M. Sun, I. Konstantelos, G. Strbac, C-vine copula mixture model for clustering of residential electrical load pattern data, IEEE Trans Power Syst 32 (3) (2017) 2382–2393, https://doi.org/10.1109/TPWRS.2016.2614366.

[21] J. Yu, K. Chen, J. Mori, M.M. Rashid, A Gaussian mixture copula model based localized Gaussian process regression approach for long-term wind speed prediction, Energy 61 (2013) 673–686 https://doi.org/10.1016/j.energy.2013.09.013.

[22] K. Mets, F. Depuydt, C. Develder, Two-stage load pattern clustering using fast wavelet transformation, IEEE Transactions on Smart Grid 7 (5) (2016) 2250–2259, https://doi.org/10.1109/TSG.2015.2446935.

[23] Z. Jiang, R. Lin, F. Yang, B. Wu, A fused load curve clustering algorithm based on wavelet transform, IEEE Trans Ind Inf 14 (5) (2018) 1856–1865, https://doi.org/10.1109/TII.2017.2769450.

[24] W. Wu, M. Peng, A data mining approach combining k-means clustering with bagging neural network for short-term wind power forecasting, IEEE Internet Things J 4 (4) (2017) 979–986, https://doi.org/10.1109/JIOT.2017.2677578.

[25] M. Ghofrani, D. Carson, M. Ghayekhloo, Hybrid clustering-time series-bayesian neural network short-term load forecasting method, In: Proc. 2016 North American power Symposium (NAPS), Denver, USA, 2016, pp. 1–5, https://doi.org/10.1109/NAPS.2016.7747865.

[26] G. Guo, L. Chen, Y. Ye, Q. Jiang, Cluster validation method for determining the number of clusters in categorical sequences, IEEE Trans Neural Netw Learn Syst 28 (12) (2017) 2936–2948, https://doi.org/10.1109/TNNLS.2016.2608354.

[27] R. Li, F. Li, N.D. Smith, Multi-resolution load profile clustering for smart metering data, IEEE Trans Power Syst 31 (6) (2016) 4473–4482, https://doi.org/10.1109/TPWRS.2016.2536781.

[28] S. Bogucharskiy, V. Mashtalir, Image segmentation via x-means under overlapping classes, In: Proc. 2015 International Scientific and Technical Conference ”Computer Sciences and information technologies” (CSIT), Yerevan, Armenia, 2015, pp. 45–47, https://doi.org/10.1109/STC-CSIT.2015.7325427.

[29] T. Erdelic, S. Vrbancic, L. Rosic, A model of speed profiles for urban road networks using g-means clustering, In: 2015 38th International convention on information and communication technology, Electronics and Microelectronics (MIPRO), 2015, pp. 1081–1086, https://doi.org/10.1109/MIPRO.2015.7160436.

[30] T. Zhang, G. Zhang, J. Lu, X. Feng, W. Yang, A new index and classification approach for load pattern analysis of large electricity customers, IEEE Trans Power Syst 27 (1) (2012) 153–160, https://doi.org/10.1109/TPWRS.2011.2167524.

[31] I.P. Panapakidis, M.C. Alexiadis, G.K. Papagiannis, Enhancing the clustering process in the category model load profiling, IET Generation, Transm Distrib 9 (7) (2015) 655–665, https://doi.org/10.1049/iet-gtd.2014.0658.

[32] D. Pelleg, A. Moore, X-means: Extending k-means with efficient estimation of the number of clusters, Mach Learn.

[33] R.L. Wasserstein, N.A. Lazar, The asa's statement on p-values: Context, process, and purpose, Am Statistician 70 (2) (2016) 129–133, https://doi.org/10.1080/00031305.2016.1154108.

[34] R. Nuzzo, Scientific method: Statistical errors, Nature 506 (2014) 150–152, https://doi.org/10.1038/506150a.

[35] Y. Xu, W. Qu, Z. Li, G. Min, K. Li, Z. Liu, Efficient k -means++ approximation with mapreduce, IEEE Trans Parallel Distrib Syst 25 (12) (2014) 3135–3144, https://doi.org/10.1109/TPDS.2014.2306193.

[36] National grid, accessed April 4, 2018. URL https://www.nationalgrid.com/uk.