From Landscape to Portrait: A New Approach for Outlier Detection in Load Curve Data

Guoming Tang, Kui Wu, *Senior Member, IEEE*, Jingsheng Lei, Zhongqin Bi, and Jiuyang Tang

***Abstract*—In power systems, load curve data is one of the most important datasets that are collected and retained by utilities. The quality of load curve data, however, is hard to guarantee since the data is subject to communication losses, meter malfunctions, and many other impacts. In this paper, a new approach to analyzing load curve data is presented. The method adopts a new view, termed *portrait*, on the load curve data by analyzing the periodic patterns in the data and reorganizing the data for ease of analysis. Furthermore, we introduce algorithms to build the virtual portrait load curve data, and demonstrate its application on load curve data cleansing. Compared to existing regression-based methods, our method is much faster and more accurate for both small-scale and large-scale real-world datasets.**

***Index Terms*—Load curve data cleansing, pattern analysis.**

NOMENCLATURE

Related to Portrait and Landscape Data

t_i	The ith timestamp.
$y(t)$	The load curve data at time t.
p_i	The ith basic portrait dataset (BPD).
P_i	The ith virtual portrait dataset (VPD).
l_i	Landscape data in the ith period.
L_i	The ith virtual landscape data (VLD).
sp_i	Span of the ith BPD or VPD.
e_i	Characteristic vector of the ith BPD or VPD.
θ_i	Median value of the ith BPD or VPD.
M_i	Median absolute deviation (MAD) of the ith BPD or VPD.
s_{ij}	Similarity between the ith and the jth portrait datasets.
d_n	The mean distance of n virtual portrait or landscape datasets.

Related to Outlier Detection

$out(\alpha, \Theta)$	The outlier region of a certain distribution with parameter vector Θ, with confidence coefficient or significance index α.
$N(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2.
$G(\beta, \gamma)$	Gamma distribution with shape parameter β and scale parameter γ.
Q_1, Q_3	The lower quartile and upper quartile of a boxplot, respectively.
IQR	The interquartile range of a boxplot (i.e., $Q_3 - Q_1$).
df	The degree of freedom in B-spline smoothing.

Manuscript received August 14, 2013; revised September 27, 2013, December 23, 2013, and February 05, 2014; accepted March 08, 2014. Date of current version June 18, 2014. This work was supported by the Natural Sciences and Engineering Research Council of Canada, National Natural Science Foundation of China (No. 61373152), and Shanghai Committee of Science and Technology, China (No. 13ZR1417500). Paper no. TSG-00665-2013.

G. Tang and K. Wu are with the Computer Science Department, University of Victoria, Victoria, BC V8W 3P6, Canada (e-mail: guoming@uvic.ca).

J. Lei and Z. Bi are with the School of Computer and Information Engineering, Shanghai University of Electric Power, Shanghai 201101, China.

J. Tang is with the Science and Technology on Information Systems Engineering Lab, National University of Defense Technology, Changsha 410111, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2014.2311415

I. INTRODUCTION

In recent smart grid research [10], [11], [14], load curve data, which refers to electric energy consumption data collected and retained by utilities, has become one of the most important datasets for a broad spectrum of applications. For electric utilities, the analysis of load curve data plays a significant role in day-to-day operations, system reliability, and energy planning. For the energy consumers, load curve data provides them with abundant information on their daily and seasonal energy cost, helping them make timely response to save expense. Overall, the importance of load curve data in the demand side management (DSM) of smart grid makes it the critical information in modern electric industry.

Due to the critical meaning of load curve data, its quality is of vital importance. Nevertheless, load curve data is subject to pollution caused by many factors, such as communication failures, meter malfunctions, unexpected interruption or shutdown of power stations, unscheduled maintenance, and temporary closure of production lines. In this paper, we call load curve data *polluted* when it significantly deviates from its regular patterns or when some data items are missing. Due to its huge volume, it would be nearly impossible to manually identify the polluted load curve data. Clearly, an efficient, automatic method is needed to solve the *load curve data cleansing* problem, i.e., to detect and fix polluted load curve data.

A. Motivation

We have observed that all existing work arranges load curve data in chronological order, i.e., the load curve data is strictly treated as a time series. As shown in Fig. 1, the hourly energy consumption of over one hundred residential houses was

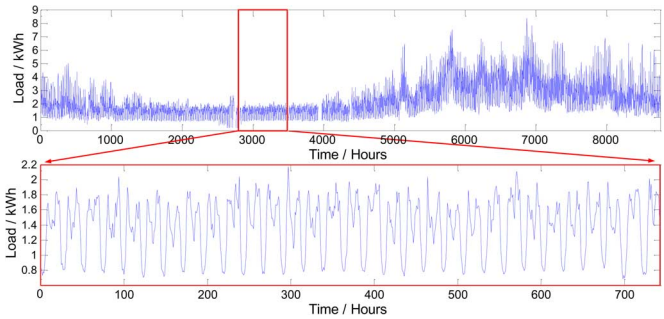

Fig. 1. Average energy consumption of 112 residential houses in the U.S. for one year from 01/04/2006 to 31/03/2007 (above) and data for one month from 01/08/2006 to 31/08/2006 (below), provided by Pacific Northwest National Laboratory [18].

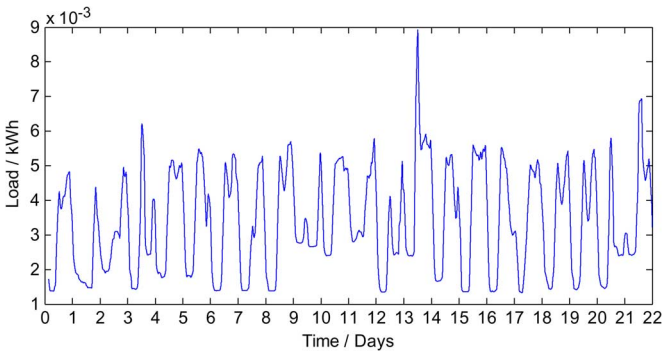

Fig. 2. Daily energy consumption of an individual residential house in Waterloo, ON, Canada, from 23/01/2011 to 13/02/2011, provided by Singh *et al.*, University of Waterloo [29].

recorded for one year (8760 hours) and displayed in the 2-D coordinate system, with x-axis representing the time and y-axis the load values (kWh). In addition to aggregated data of multiple households, the low-level energy consumption of an individual household is illustrated in Fig. 2, which shows the similar periodic pattern.[1] We call such type of arrangement of load data as *landscape data*. Landscape data is easy to understand, but it poses several barriers to efficient analysis.

- First, in a short time window (say 1 to 2 hours), the correlation between time and the load values may be hard to capture due to two reasons: 1) some random events may play a dominant role in electric load; and 2) it is hard to obtain a unified model to capture the local pattern, which may change over time.
- Second, in a relatively long time (say days), even though certain regular patterns of the load curve can be found, the load curve along the timeline is nonlinear and may be too complicated to model with fixed parameters.
- Third, with landscape data, each sample is usually treated equally, making it difficult to effectively capture special behavioral features. For instance, the energy consumption for a cafeteria is low and stable when it is closed and high during breakfast and lunch times. In this sense, it would be better if load data could be treated differently during the former period (say from 7:00 pm to 7:00 am) and during the latter periods (say from 7:00 am to 9:00 am and from 11:00 am to 1:00 pm).

[1]Note that some low-level household load curve data may not have periodic pattern at all. In this case, the method developed in this paper does not bring benefit.

B. Our Contribution

Based on the above observations, we challenge the traditional landscape data as an efficient way to organize load curve data. The following contributions are made in the paper:

- A new view, called *portrait*, is proposed for load data analysis. Switching perspective from landscape to portrait, some hidden behavioral patterns in the load data become prominent, such as the numerical stability of load curve data in the same hours of different days.
- With Fourier analysis, an algorithm is designed to *automatically* transform a landscape data to portrait data. We further extend the method to build *virtual portrait datasets*, meaning of which will be disclosed later, to address the problem in the third observation raised above.
- A data preprocessing method is proposed, so that nonstationary load data can be effectively handled with the help of virtual portrait datasets.
- Efficient algorithms are designed to use virtual portrait data for both small-scale and large-scale load data cleansing. Our experimental results show that our portrait based method is faster and more accurate, compared to the state-of-the-art regression-based methods.

II. Related Work

Load curve data cleansing in smart grid has caught more and more attention recently, from both academia and industry. So far, most related work considers the polluted data as outliers in load pattern and focuses on outlier detection.

Regression-based methods have been widely studied for outlier detection in time series [1], [10], [23], [25]. In [10], a non-parametric regression method based on B-spline and kernel smoothing was proposed and applied to identify polluted data. In [1], the residual pattern from regression models was analyzed and applied to construct outlier indicators, and a four-step procedure for modeling time series in the presence of outliers was also proposed. Greta *et al.* [23] considered the estimation and detection of outliers in time series generated by a Gaussian auto-regression moving average (ARMA) process, and showed that the estimation of additive outliers was related to the estimation of missing observations. The ARMA model was also utilized in [1], [2], [16], [28] as the basic model for outlier detection. In general, the regression-based methods are established on empirical knowledge and their parameters are regulated manually according to the domain knowledge of experts. As a result, such methods are subject to either underestimation or overestimation.

Since load curve data consists of one-dimensional real values, univariate statistical methods can deal with outliers in such dataset [12], [13], [15], [17]. Most univariate methods for outlier detection assume that the data values follow an underlying known distribution. Then, the outlier detection problem is transformed to the problem of finding the observations that lie in a so-called outlier region of the assumed distribution [13]. Even though those methods have been proved simple and effective, we may not always know the underlying distribution

of the data. This is unfortunately true for load curve data, e.g., the distribution of the data shown in Figs. 1 and 2 is unknown.

In addition to the above methods, data mining techniques have also been developed to detect outliers, such as k-nearest neighbor [21], [27], k-means [3], [26], k-medoids [6], density-based clustering [22], etc. In general, these methods classify the observations with similar features, and find the observations that do not belong strongly to any cluster or far from other clusters. Nevertheless, most data mining techniques are designed for structured relational data, which may not align well for the need of outlier detection in load curve data. In addition, these methods are normally time consuming because they need a training process on a large dataset.

III. Introduction of Portrait Data

A. Portrait Data

Definition 1: Consider a periodic function $f(x)$ with period of T defined over $[0, NT]$. We split one period of time $[0, T]$ into n even slices, i.e., $0 = x_0 < x_1 < x_2 \ldots < x_n = T$. The **portrait data** of function $f(x)$ corresponding to the ith time slice $(0 \le i \le n)$, denoted by p_i, is defined as the dataset

$$p_i := \{f(x) | x \in [x_i + kT, x_{i+1} + kT], 0 \le k \le N\}. \quad (1)$$

Definition 2: The **span** of a portrait data p_i is defined as

$$sp_i := x_{i+1} - x_i. \quad (2)$$

Similarly, for discrete periodic load curve data with even spacing labeled as $\{y(0), y(1), y(2), \cdots\}$, the portrait data are composed with the data points falling within the corresponding time intervals, i.e., the portrait data p_i is constructed as

$$p_i := \{y(t) | t = t_i + kT, 0 \le k \le N\}. \quad (3)$$

B. Example of Portrait Data

To help better understand the portrait data, we use the one-month load curve data in Fig. 1 as an example to illustrate portrait data visually.

Noticing that the data exhibits a periodicity of 24 hours, we divide the original time line by 24 hours into 31 slices (days) and rearrange the slices in parallel. In this way, we transform the 2-D landscape data into 3-D space, with the x-axis representing hours, the y-axis days, and z-axis the load values, as shown in Fig. 3. To view the energy consumption of each hour in the 31 slices, we rotate the figure in the x-y coordinate by 90 degrees, and redraw the data into 24 slices. Each slice represents a portrait data consisting of the energy consumption at the same hour in the 31 days, as shown in Fig. 4. Immediately, we can observe that: *the values in each portrait dataset are relatively stable*.

C. Characteristic Vector of Portrait Data

Intuitively, a portrait dataset should include values with a very small variation. There are many ways to model this phenomenon. In this paper, we use the following characteristic vector to describe the portrait data.

Definition 3: The **characteristic vector** of portrait data p_i is defined as

$$e_i := [\theta_i, M_i] \quad (4)$$

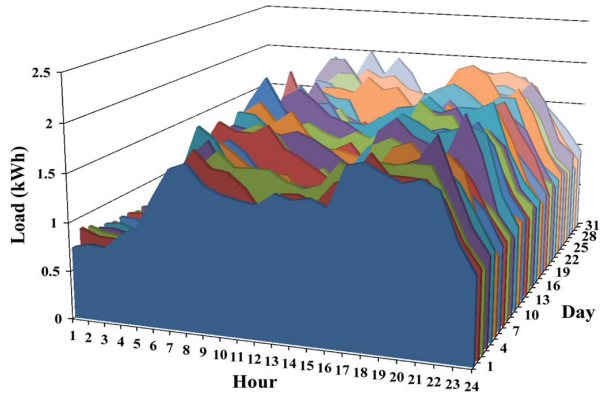

Fig. 3. Divide timeline into 31 pieces by 24 hours and reposition the pieces in parallel.

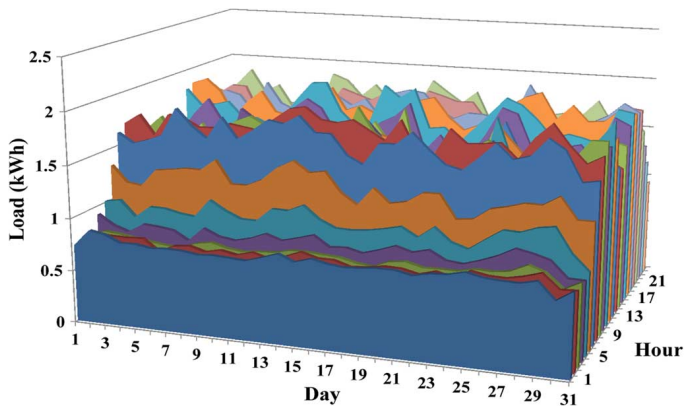

Fig. 4. Switch the view to portrait.

TABLE I
Characteristic Vectors of Portrait Data of the First 10 Hours in Fig. 4, Compared With Landscape Data (Unit: kWh)

hr.	1	2	3	4	5	6	7	8	9	10	1-10
θ	0.79	0.78	0.77	0.84	0.99	1.30	1.69	1.76	1.69	1.60	1.14
M.	**0.04**	**0.01**	**0.02**	**0.04**	**0.05**	**0.08**	**0.09**	**0.07**	**0.09**	**0.11**	**0.42**

where θ_i and M_i represent the median and the median absolute deviation (MAD) of values in p_i, respectively.

Since the data may be contaminated by outliers, we use the *median* and *MAD* instead of *mean* and *standard deviation* to represent central tendency and statistical dispersion of a portrait dataset, respectively. The *median* and *MAD* are more robust measures [19].

For the data in Fig. 4, the characteristic vectors of portrait data of the first 10 hours are summarized in Table I. The last column of the table shows that the *MAD* value of landscape data is significantly higher. The results indicate that each portrait dataset is much more stable than the landscape data.

Definition 4: The **similarity** of two portrait datasets p_i,p_j with characteristic vectors e_i, e_j, respectively, is defined as

$$s_{ij} := \begin{cases} \infty, & \text{if } e_i = e_j \\ \frac{1}{\|e_i - e_j\|_2}, & \text{otherwise.} \end{cases} \quad (5)$$

We can develop heuristic algorithms to merge multiple portrait datasets with a high similarity into a virtual portrait dataset, which will be introduced in detail in Section IV.

D. Properties of Portrait Data

Compared to landscape data, the portrait data has the following desirable properties:

- The data values within the same portrait are similar and can be processed together even if they are separated in the original time domain.
- The data values within the same portrait dataset can be captured with a simple model, for which numerous fast data cleansing methods can be applied. In contrast, landscape data is normally nonlinear and requires complicated nonlinear regression-based methods.
- With portrait data, users' behavioral patterns in different time periods can be modeled. In Fig. 4, the energy consumption in the first hour of each day is quite stable and low, but the situation for the seventh hour is quite different. As such, a data point with small deviation in the first slice should be captured as an outlier, but may be considered regular in the seventh slice. In this way, we can improve the accuracy of outlier detection.

It is worth noting that portrait data is *not* just a data visualization trick. It is helpful to design efficient algorithms for load curve data analysis and cleansing. In specific, due to the stability in each portrait data, it is much easier to build simple models to capture the outliers. In addition, by combining similar portrait slices into one virtual slice, we can build *virtual portrait* dataset, which further speeds up data processing.

IV. Construction of Portrait Data

A. Analysis of Periodic Pattern in Landscape Data

In order to *automatically* construct portrait data, we need to find out the time period of landscape load curve data. In our daily life, the energy consumption of different houses or buildings is usually periodic, either hourly, daily or weekly. When the volume of landscape data is big, an automatic method is needed to quickly discover the periodic behavior hidden in the landscape data. In this paper, Fourier analysis [5] is used for this purpose.

According to Fourier transform, given a nonsinusoidal periodic function

$$f(t) = f(t + kT), \quad k = 0, 1, 2, \cdots \tag{6}$$

if in one cycle of the periodic function there are finite maximum and minimum values, as well as the finite number of first category discontinuous points,[2] the function can be unfolded into a convergent Fourier Series, i.e.,

$$f(t) = A_0 + A_1 cos(\Omega t + \psi_1) + \sum_{k=2}^{\infty} A_k cos(k\Omega t + \psi_k) \tag{7}$$

where A_0 is called the *constant component* and $A_1 cos(\Omega t + \psi_1)$ the *fundamental component*. The frequency of the fundamental component discloses the lowest frequency in the original function $f(t)$, which can be used to construct the portrait data.

Since the load curve data is discrete, we should use another form for Fourier transform, discrete Fourier transform (DFT), to convert a finite list of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids, ordered by their frequencies. To speed up the process, fast Fourier transform (FFT) is adopted, which is developed upon DFT and works much faster.

In practice, the sampling interval for residential energy consumption on the utility side is normally 15 minutes [9]. Considering the periodic pattern in load curve is relatively longer, such as one day (24 hours), the sampling rate is high enough to acquire the time period of load curve data.

B. Construction of Basic and Virtual Portrait Data

The next step is to decide how many slices of portrait data should be split.

One solution is to split the load curve data with the span of sampling interval, which will result in portrait data with the highest resolution. However, since the sample rate may be significantly high, such kind of splitting may result in too many portrait data slices. Considering that the characteristic vectors of some portrait datasets are similar, we merge them together into a virtual portrait dataset to speed up data cleansing. Therefore, a two-phase method is developed.

1) Build Basic Portrait Datasets: The portrait datasets obtained in this phase are called *basic portrait dataset* (BPD). With FFT, the fundamental period of load curve data can be obtained. Assuming that there are r samples in one period, we can obtain r basic portrait datasets $\{p_0, p_1, \cdots, p_r\}$. Accordingly, we can calculate the characteristic vector of each basic portrait dataset, denoted by $\{e_0, e_1, \cdots, e_r\}$, respectively.

2) Build Virtual Portrait Datasets: We merge multiple basic portrait datasets with similar characteristic vectors into one *virtual portrait dataset* (VPD). As such, A clustering algorithm is needed to partition the basic portrait datasets into exclusive clusters such that within each cluster, the pairwise similarity of basic portrait datasets is no less than a given threshold. In order to accelerate data analysis, it is desirable to minimize the total number of clusters. This optimization problem can be formulated as follows:

- **Input**: Basic portrait data $\{p_1, p_2, \cdots, p_r\}$ and their corresponding characteristic vectors $\{e_1, e_2, \cdots, e_r\}$. A given threshold s_0 on similarity.
- **Output**: Minimum number of virtual portrait datasets, denoted by $\{P_1, P_2, \cdots, P_n\}$ such that within each virtual portrait dataset, the pairwise similarity of the basic portrait datasets is no less than s_0.

$$\begin{aligned} &\underset{\{P_1,P_2,\cdots,P_n\}}{\textbf{minimize}} \quad n \\ &\textbf{subject to} \\ &\qquad \bigcup P_i = \{p_1, p_2, \cdots, p_r\} \\ &\qquad P_i \bigcap P_j = \varnothing, i \neq j \\ &\qquad P_i = \{\{p_{l_1}, p_{l_2}, \cdots, p_{l_m}\} \,|s_{l_s l_t} \geq s_0\} \\ &\qquad 1 \leq i, j \leq n; 1 \leq m \leq r; 1 \leq l_s, l_t \leq m \end{aligned} \tag{8}$$

In order to solve the above problem, a graph $G = (V, E)$ is constructed, where each vertex $v \in V$ represents a BPD

[2] A discontinuous point x is called the first category discontinuous point where there exist finite limits from the left $f(x-0)$ and from the right $f(x+0)$ for f.

and an edge is built between two vertices if their similarity is no less than s_0. It is easy to see that the problem is equivalent to the *clique-covering* problem, which has been proven to be NP-hard [24]. Hence, a *greedy clique-covering algorithm* is adopted to obtain an approximate solution. Algorithm 1 shows the pseudo code of the greedy clique-covering problem.

Algorithm 1 Greedy Clique-Covering Algorithm

Input: Graph $G = (V, E)$
Output: A set of cliques P that completely cover G
1: Initialize uncovered vertex set $V' \leftarrow V$
2: Initialize number of cliques, $n = 0$
3: **while** $V' \neq \Phi$ **do**
4: $n = n + 1$
5: Find $v \in V'$ with the highest node degree
6: Find $U \subseteq V'$ with $u \in U$ and $(u, v) \in E$
7: Construct subgraph $G' = (U, D)$ where U includes all vertices adjacent to v, and D includes the associated links
8: Initialize clique $P_n = \{v\}$
9: **for** each $w \in U$ **do**
10: **if** w is adjacent to all vertices in P_n **then**
11: $P_n \leftarrow P_n \cup \{w\}$
12: **end if**
13: **end for**
14: $V' \leftarrow V' \setminus P_n$
15: **end while**
16: **return** $P_1, P_2, \cdots, P_n$

The basic idea of the algorithm is to find cliques that cover more vertices that have not been clustered. Heuristically, the vertices with larger degrees may have a better chance of resulting in a smaller number of cliques. Thus, the search starts from the vertex with the largest degree, until all vertices are covered. Obviously, a resulted cluster is a clique in the graph. Since each vertex represents a BPD, a clique represents a VPD.

Lemma 1: The computational complexity of Algorithm 1 is lower bounded by $O(r \log r)$ and upper bounded by $O(r^2 \log r)$, where r is the number of basic portrait datasets.

Proof: Since the similarity of two basic portrait datasets is calculated with their characteristic vectors consisting of two values, the graph G in Algorithm 1 is actually a geometric graph in the 2-D plane. Any clique resulted from Algorithm 1 can be bounded by some rectangle region in the 2-D plane. According to [20], the largest clique of a rectangle intersection graph can be found with computational complexity no more than $O(r \log r)$. Since in Algorithm 1 the number of iterations in finding cliques could range from 1 to r, the computational complexity ranges from $O(r \log r)$ to $O(r^2 \log r)$. ■

V. Load Curve Data Cleansing

In this section, we show portrait data can help load curve data cleansing. Load curve data cleansing involves two phases: 1) detecting outliers and 2) fixing the missing or aberrant values in the dataset.

A. Detection of Outliers

Formally, for a given distribution F, the outlier detection problem is to identify those values that lie in a so-called *outlier region* defined below:

Definition 5: For any confidence coefficient $\alpha, 0 < \alpha < 1$, the α-**outlier region** of F distribution with parameter vector Θ can be defined by

$$out(\alpha, \Theta) = \left\{x : x < Q_{\frac{\alpha}{2}}(\Theta) \text{ or } x > Q_{1-\frac{\alpha}{2}}(\Theta)\right\} \quad (9)$$

where $Q_q(\Theta)$ is the q quantile of function $F(\Theta)$.

Since we usually do not have *a priori* knowledge on the distribution of portrait data, various possible cases should be considered. Note that performing statistical test to find out the distribution of load curve data does not work well when the load data is polluted. We need to consider several potential cases for outlier detection.

1) Case 1: Outlier Detection for Normal Distributed Data: The normal distribution can be adopted as an empirical distribution, which has been proved to be effective in general situations [4].

According to (9), for a normal distribution $N(\mu, \sigma^2)$, its α-*outlier region* is

$$out\left(\alpha, (\mu, \sigma^2)\right) = \left\{x : |x - \mu| > \Phi_{1-\frac{\alpha}{2}}\sigma\right\} \quad (10)$$

where Φ_q is the q quantile of $N(0, 1)$. For normal distributed portrait datasets $P_i, i = 1, 2, \cdots$, we claim that a value x is an α-outlier in P_i, if $x \in out(\alpha, (\widehat{\mu_i}, \widehat{\sigma_i}^2))$, where $\widehat{\mu_i}$ and $\widehat{\sigma_i}$ are unbiased estimators of μ_i and σ_i, respectively. Since the data may be contaminated by outliers, we use the *median* and *MAD* instead of *mean* and *standard deviation* in our later detection.

2) Case 2: Outlier Detection for Gamma Distributed Data: It has been shown that the aggregated residential load at a given time instant follows the gamma distribution [7], [8]. In this light, the gamma distribution is also a good candidate distribution for outlier detection.

According to (9), for a gamma distribution with shape parameter β and scale parameter γ, $G(\beta, \gamma)$, its α-*outlier region* is

$$out\left(\alpha, (\beta, \gamma)\right) = \left\{x : x < F^{-1}_{\frac{\alpha}{2}}(\beta, \gamma) \textbf{ or } x > F^{-1}_{1-\frac{\alpha}{2}}(\beta, \gamma)\right\} \quad (11)$$

where F^{-1} is the inverse cumulative distribution function of $G(\beta, \gamma)$, and $F^{-1}_q(\beta, \gamma)$ is the q quantile of $G(\beta, \gamma)$.

If we assume that virtual portrait datasets, $P_i, i = 1, 2, \cdots$ follow a gamma distribution $G(\beta, \gamma)$, we can use (11) for outlier detection. In this case, $(\widehat{\mu_i}^2/\widehat{\sigma_i}^2)$ and $(\widehat{\sigma_i}^2/\widehat{\mu_i})$ are the moment estimators of β and γ, respectively.

3) Case 3: Outlier Detection for Small-Size Portrait Data: In the above outlier detection strategies, the size of portrait datasets is assumed to be large. Otherwise, the parameter estimation may be inaccurate. When the size of samples is small, Tukey *et al.* [30] introduce a graphical procedure called *boxplot* to summarize univariate data.

The boxplot uses median and lower and upper quartiles (defined as the 25th and 75th percentiles). If the lower quartile is Q_1 and the upper quartile is Q_3, then the difference $(Q_3 - Q_1)$ is called interquartile range or IQR. After arranging data in order,

the ones falling in the following outlier region are identified as outliers:

$$out\,(\rho,(Q_1, Q_3)) = \{x : x < Q_1 - \rho \cdot IQR \textbf{ or } x > Q_3 + \rho \cdot IQR\} \tag{12}$$

where ρ is an index of significance, and the outliers are said to be "mild" when $\rho = 1.5$ and "extreme" when $\rho = 3$.

Overall, the above three cases cover most situations a user may meet in portrait load data cleansing. Nevertheless, other strategies can also be chosen as long as they give more precise model of the portrait data.

B. Replacing Missing Data or Aberrant Data

We mainly focus on outlier detection in this paper for two reasons.

1) Imputation of missing data can be easily done after we obtain the characteristic vector of portrait data, e.g., we could replace a missing value with the median of the corresponding portrait dataset.
2) Replacing aberrant values requires human interaction, since it needs the user to further confirm whether or not an outlier is a corrupted value. The user can either 1) replace the outlier with an acceptable value of the corresponding dataset, e.g., the mean value for Case 1 and Case 3 and the value of β/γ for Case 2, or 2) leave the outlier unchanged, as long as the cause of creating the outlier can be explained, such as the stimulation of holidays/special events.

Note that data imputation is normally carried out after outlier detection. It is common to initially set the missing data to default values of zeros, which are likely to be outliers and then are replaced with acceptable values. This strategy has been used in [10]. Nevertheless, the default value can be altered by the user according to different scenarios. For instance, if there exist valid load values close to zero, we can set the default value for missing data to a very large value, so that missing data can be identified easily as outliers.

If the user has explicitly learned the cause of the aberrant or missing data and found any above or other replacing approach that fits the needs well, the approach can be incorporated with our outlier detection approach and works automatically.

VI. Handling Nonstationary Landscape Data

The construction and cleansing of portrait data are based on the assumption that the landscape data are stationary along the timeline. Informally, a stationary time series has a well-defined mean around which it can fluctuate with constant finite variance. This assumption may be true during a short time period, while in a significantly long time period, such as one year, the load curve shows seasonal patterns and is usually not stationary. For the one year load values shown in Fig. 1, they are not stationary along the whole timeline. Consequently, the portrait data cleansing strategies in Section V may not work well.

To deal with this problem, a preprocessing method is proposed, based on two observations: 1) the load curve data exhibits periodicity in a small time scale (e.g., one month) and 2) the fundamental period of load curve data (e.g., one day) in different small time windows (e.g., January and June) is (nearly) the same. Both observations will be further validated in our late experimental evaluation.

Within a small time-scale (e.g., several days to one month), the fundamental period of the landscape data can be obtained via FFT. We first divide the whole time with the length of the fundamental period, and use the data within each time period as the basic building block. For the landscape data in the ith period, which is denoted as l_i, we define its *characteristic vector* $e_i := [\theta_i, M_i]$, where θ_i and M_i represent *median* and *median absolute deviation (MAD)* of the values in l_i, respectively. Thus, similar to portrait data, the *similarity* of the landscape data in two different period, l_i and l_j, can be defined as $s_{ij} := 1/\|e_i - e_j\|$, or ∞ if $\|e_i - e_j\| = 0$. Here we slightly abuse the notation by using the e_i and s_{ij} to denote the characteristic vector and similarity, respectively, for both landscape data and portrait data. Their meaning, however, is easy to figure out from the context.

For the landscape data of different periods that have similar characteristic vectors, we merge them into one dataset, which is called a *virtual landscape dataset* (VLD). If the whole landscape data consists of n (non-overlapping) periods, the problem of constructing its VLDs can be formally defined as follows.

- **Input**: Landscape data $\{l_1, l_2, \cdots, l_n\}$ and their characteristic vectors $\{e_1, e_2, \cdots, e_n\}$. A given similarity threshold s_0.
- **Output**: Minimum number of VLDs $\{L_1, L_2, \cdots, L_m\}$, $m \ll n$.

Note that the above problem is exactly the same with Problem (8). Thus, Algorithm 1 can be reused to construct VLDs. Since all data points in each of the VLDs have similar properties, they are stationary and meet the requirement for portrait data construction and cleansing. We can then further build corresponding portrait data for each VLD.

VII. Implementation and Experimental Evaluation

In this section, the real-world trace data shown in Fig. 1 is used to construct virtual portrait datasets. We implement the multiple strategies introduced above to detect outliers, and perform numerous experiments to evaluate the performance.

So far, there is limited literature on data cleansing applications to smart grid, and one significant contribution was in [10], in which a non-parametric regression method based on B-spline smoothing was proposed to help users identify outliers. For comparison purpose, we implement the B-spline smoothing method and compare it with our own method.

A. Fundamental Period

By applying FFT on the landscape data, we got the frequency spectrum of landscape data, and the frequency of the second peak corresponds to the fundamental frequency. Its reciprocal is the fundamental period of the landscape data. After calculation, the fundamental frequency of the landscape data in Fig. 1 is 1.1574×10^{-5}, which precisely results in a period of 24 hours (86 400 seconds).

In addition, a sensitivity experiment is made with 1000 tests on the data. In each of the test, a random time period longer than 1 month but shorter than 1 year was chosen. According to the results, the mean value of the identified periods is 23.9984 hours with variance of 1.9952×10^{-4}. Therefore, we can conclude that

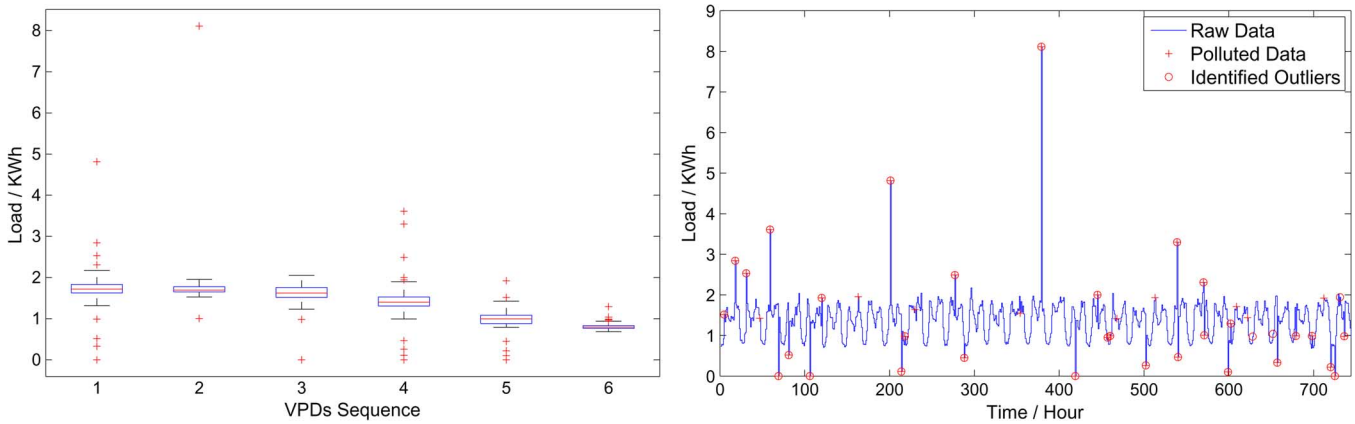

Fig. 5. Result of outlier detection from IQR-based virtual portrait data cleansing.

the accuracy of identified fundamental period is not sensitive to the time period and the starting time of the samples.

B. The Optimal Threshold Value

By applying Algorithm 1, a number of virtual (portrait/landscape) datasets can be built for a given threshold value on the similarity measure. By changing the threshold value from small to large, we can get a series number of virtual datasets. We are thus faced with the following question: what is the optimal threshold value?

In order to answer the above question, *mean distance* is defined to estimate the "quality" of virtual datasets (i.e., whether or not two virtual datasets are clearly separate). For n virtual datasets with corresponding characteristic vectors $e_1, e_2, \ldots, e_n$, the mean distance is defined as

$$d_n := \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{s_{ij}}{\binom{n}{2}} \tag{13}$$

where s_{ij} is defined by (5). Obviously, with the same number of virtual datasets, the larger the d, the clearer the separation among the virtual datasets.

By changing the threshold value on the similarity measure, we can obtain different numbers of virtual datasets. Applying the *ELBOW criterion*[3] [31], we can get the optimal number of virtual datasets. The optimal threshold on the similarity measure is thus the one that leads to this number of virtual datasets.

C. Performance Metrics

In outlier detection, four statistical indicators are widely used: 1) true positive (TP), the number of points that are identified correctly as outliers; 2) false positive (FP), the number of points that are normal but are identified as outliers; 3) true negative (TN), the number of points that are normal and are not identified as outliers; 4) false negative (FN), the number of points that are outliers but are not identified. Using TP, FP, TN, and FN, we evaluate the following four broadly used performance metrics: accuracy, precision, recall, and F-measure. Accuracy is the degree of closeness of measurement to the actual situation as a whole; precision is the percentage of correctly detected corrupted regions with regard to the total detected regions; recall is the percentage of correctly detected regions with regard to pre-labeled corrupted regions; the F-measure is a harmonic mean of precision and recall, i.e.,

$$\text{F-measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{14}$$

Furthermore, running time (R.T.) and memory usage (M.U.) of program are used to measure the time and space consumption of different methods, respectively. We implement them in R and test them with 32-bit Windows OS with 3.4-GHz CPU and 4-GB RAM.

The real-world data shown in Fig. 1 is used for the evaluation. Since this dataset is relatively clean, we ask three students to distort the data with "falsification," i.e., they are asked to arbitrarily modify the load curve data within the range of $[0, \infty)$. Five percent of the samples are changed and labeled.

In our tests, the confidence coefficient is set as $\alpha = 0.05$, which results in a confidence interval of 95%. Besides, in the IQR-based method, ρ is set as 1.5, and in the B-spline smoothing method, the degree of freedom (df) is treated as a variable and trained when smoothing the load curve.

D. Results From Small-Scale Data

The one-month data from 01/08/2006 to 31/08/2006 in Fig. 1 are first used for evaluations. Since these data are stationary, virtual portrait data construction strategy is applied directly. Six virtual portrait datasets are resulted based on the optimal threshold value.

In addition to the strategies introduced in Section V, the B-spline smoothing method is also applied for each virtual portrait dataset. Performance metrics are computed with the outcomes from different methods, and the final results are summarized in Table II. Furthermore, an outcome from IQR-based portrait data cleansing is shown in Fig. 5.

From the above results, we can find that our virtual portrait data cleansing strategies perform much better than B-spline smoothing. For this dataset, both gamma distribution based and IQR-based detection methods perform better than the normal distribution based one. It is interesting to see that applying B-spline smoothing method to virtual portrait data does not bring clear improvement (refer to the results in fifth column of Table II). This implies that using simpler methods on portrait

[3]The concept of VPD is in principle the same as clustering. The ELBOW criterion means that we should choose a number of clusters so that adding another cluster would not model the data much better.

TABLE II
PERFORMANCE ON SMALL-SCALE DATA: VIRTUAL PORTRAIT DATA CLEANSING VERSUS B-SPLINE SMOOTHING

	Virtual Portrait Data Cleansing Strategies				**B-spline Smoothing**				
	Normal-based	Gamma-based	IQR-based	B-spline(df = 68)	df = 148	df = 188	df = 228	df = 258	df = 318
Accuracy	0.9878	0.9865	0.9879	0.9823	0.9582	0.9716	0.9715	0.9724	0.9748
Precision	0.8857	0.9375	0.9118	0.7500	0.8182	0.7917	0.7308	0.5405	0.4151
Recall	0.7750	0.7500	0.7750	0.6750	0.2250	0.4750	0.4750	0.5000	0.5500
F-measure	0.8267	0.8333	0.8378	0.7105	0.3529	0.5938	0.5758	0.5195	0.4731
$R.T.(second)$	0.036	0.036	0.037	0.021	0.040	0.051	0.075	0.089	0.118
$M.U.(MB)$	0.067	0.067	0.067	1.427	2.933	3.605	4.327	4.890	5.980

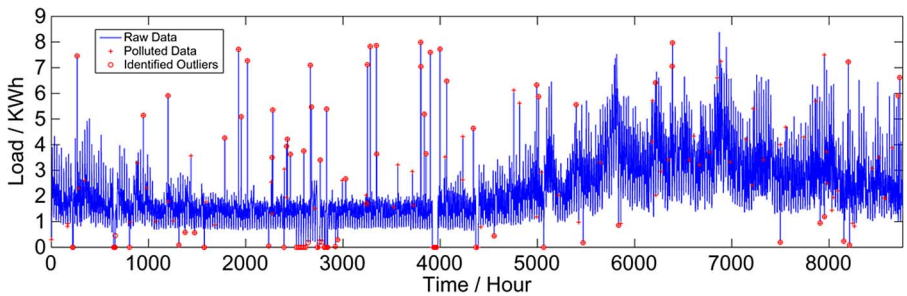

Fig. 6. Result of outlier detection from gamma distribution based virtual portrait data cleansing (7 VLDs).

TABLE III
PERFORMANCE ON LARGE-SCALE DATA: VIRTUAL PORTRAIT DATA CLEANSING VERSUS B-SPLINE SMOOTHING

	5 Virtual Landscape Datasets			**7 Virtual Landscape Datasets**			**10 Virtual Landscape Datasets**			**B-spline Smoothing**	
	N-based	G-based	I-based	N-based	G-based	I-based	N-based	G-based	I-based	df = 2815	df > 2815
Accuracy	0.9931	0.9950	0.9927	0.9939	0.9950	0.9930	0.9936	0.9951	0.9947	0.9792	–
Precision	0.6154	0.7080	0.7143	0.5820	0.7080	0.7590	0.6939	0.6864	0.7091	0.3378	–
Recall	0.5161	0.6452	0.4839	0.5726	0.6452	0.5081	0.5484	0.6532	0.6290	0.3620	–
F-measure	0.5614	0.6751	0.5769	0.5772	0.6751	0.6087	0.6126	0.6694	0.6667	0.3495	–
$R.T(second)$	2.74	2.67	2.41	3.29	3.30	3.44	4.69	4.18	4.35	74.44	–
$M.U(MB)$	0.42	0.42	0.42	0.42	0.42	0.42	0.42	0.42	0.42	595.26	–

data can achieve good performance already. It is unnecessary to use complex approaches such as B-spline smoothing on portrait data.

Furthermore, we can see that the virtual portrait data cleansing runs faster and uses much less memory than B-spline smoothing. In fact, most time and memory spent in our strategies are on the construction of virtual portrait datasets, and the overhead of data cleansing over portrait data is negligible. B-spline smoothing, however, spent over 99% of the running time and memory on the calculation of basis functions, which are used to fit the landscape load curve data.

E. Results From Large-Scale Non-Stationary Data

In practice, the size of load curve data is usually large and covers a time period as long as several years. Therefore, we also test the performance of our method on the one-year data shown in Fig. 1.

Note that the landscape data are not always stationary during the whole time window, so we preprocess the data with the method introduced in Section VI. For comparison, three solutions with 5, 7, and 10 VLDs are provided for tests and evaluations (7 VLDs are resulted from the optimal threshold value). Then for each VLD of each solution, following the same operations for small-scale dataset, we construct its virtual portrait datasets and apply portrait data cleansing strategies to identify outliers.

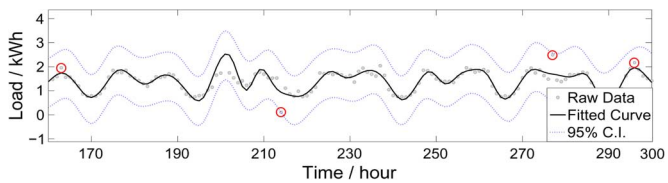

Fig. 7. Under-fitted B-spline smoothing ($df = 100$).

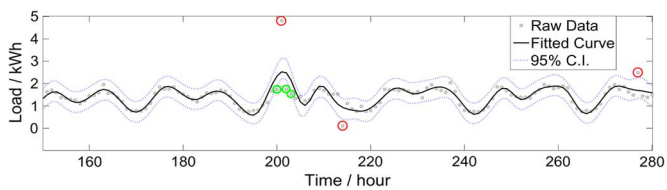

Fig. 8. Over-fitted B-spline smoothing ($df = 200$).

The results are summarized in Table III, and an outcome from 7 VLDs and gamma distribution based portrait data cleansing is shown in Fig. 6.

From the results in Table III, we can see that with nonstationary landscape data, virtual portrait data cleansing strategies are still effective and perform well. According to *F-measure*, gamma distribution based cleansing strategy does better than the other two, indicating that it achieves a good balance between precision and recall and has a better overall performance; IQR-based cleansing does better at *precision*, indicating that this strategy performs well at exactness of outlier detection.

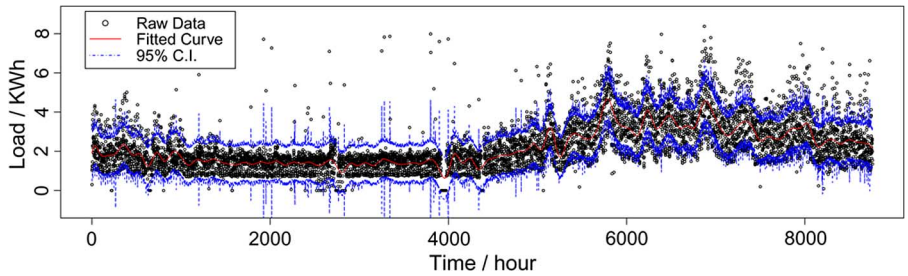

Fig. 9. Results of B-spline smoothing for large-scale data ($df = 100$).

In contrast, outlier detection with B-spline smoothing performs poorly. With the largest degree of freedom that the computation allows,[4] the *precision*, *recall*, and *F-measure* of outlier detection are all below 50%. To be worse, the overhead on running time and memory consumption is significantly higher than our method.

As shown in Fig. 6, our method does not identify many polluted data from time 5000 until the end. This is because those artificially polluted data are at a comparable level in value as the nearby regular load data. These "outliers" are similar to regular values and cannot be effectively detected with any method.

F. Discussion: Why Does B-spline Smoothing Not Perform Well on Load Curve Data?

1) Local View: To further investigate why B-spline smoothing does not perform well in load curve data cleansing, we first study the performance in a smaller, local scale. We analyze two situations shown in Figs. 7 and 8, where B-spline smoothing either under-fits or over-fits the load curve data.

From Fig. 7, we can see that the four labeled polluted data were not identified due to the under-fitted regression of B-spline. To alleviate the problem, we may increase the degree of freedom (df), but doing so may result in over-fitting. As shown in Fig. 8, in order to fit some outliers (the red dots in the figure), the fitted curve actually deviates from regular data points (the green dots in the figure), which ends up with bad performance in outlier detection. During the process from under-fitting to over-fitting, there must exist a df value which results in the best performance, but finding the best df value is time consuming.

The above phenomenon is caused by the inherent problem in regression method, as it treats each data point in the same way and tries to reduce the total estimation error. This may not work well because load curve data at different times follow different statistical features. Using the *portrait* data, in contrast, we can divide data into different groups according to their attributes, and analyze each group separately. Thus "pathological" data values may infect landscape data on a large time window but has only limited impact on portrait data. This is the essential point where B-spline smoothing performs poor while virtual portrait data cleansing does better.

2) Global View: An outcome from B-spline smoothing with $df = 100$ is shown in Fig. 9, in which we can have a global view of its performance.

From 3920h to 3975h in the load curve, the data are lost and are treated as zeros during B-spline smoothing. We can find that most missing data are not identified. This is caused by an apparent curvature trend to fit the filled data. Even if we replace the missing data with other constants, such a curvature trend is inevitable. This exposes another drawback of regression-based outlier detection methods: they cannot deal with *consecutively* polluted data. In contrast, our portrait data cleansing strategies do not have such a problem. Since all the landscape data will be separated into different portrait data, the consecutively polluted data will be evaluated and handled differently. As a result, one polluted data will not affect nearby ones.

In some special time periods such as holidays, the load values may be consecutively higher or lower in the landscape data. This scenario is similar to the above case. With regression-based outlier detection, there will be an inevitable curvature trend to fit the irregular data, while with our strategies, such data values are separated into different virtual portrait datasets and can be detected with a high possibility.

[4]B-spline smoothing with degree of freedom larger than 2815 is beyond the capability of our desktop computers.

VIII. Conclusion

A new approach was presented to organizing and analyzing load curve data. This approach was based on the inherent periodic patterns in the load curve data and reorganized the data into virtual portrait datasets that could be captured with simple models. Compared to existing regression-based analysis, portrait data based approach significantly simplified many data analysis tasks such as outlier detection. In addition, with simple data preprocessing, our method could effectively handle large-scale nonstationary load curve data. We tested our approach with real-world trace data, including a small-scale stationary dataset and a large-scale nonstationary dataset. The experimental results demonstrated that our approach was much more effective and efficient than existing regression-based methods over both small-scale and large-scale load curve data.

References

[1] B. Abraham and A. Chuang, "Outlier detection and time series modeling," *Technometrics*, vol. 31, no. 2, pp. 241–248, 1989.
[2] B. Abraham and N. Yatawara, "A score test for detection of time series outliers," *J. Time Ser. Anal.*, vol. 9, no. 2, pp. 109–119, 1988.
[3] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang, "Topic detection and tracking pilot study final report," in *Proc. Broadcast News Transcript. Understand. Workshop*, 1998, DARPA.
[4] I. Ben-Gal, *Outlier Detection. In Data Mining and Knowledge Discovery Handbook*. New York, NY, USA: Springer, 2005, pp. 131–146.
[5] P. Bloomfield, *Fourier Analysis of Time Series: An Introduction*. New York, NY, USA: Wiley, 2004.
[6] R. J. Bolton and D. J. Hand *et al.*, "Unsupervised profiling methods for fraud detection," *Credit Scoring and Credit Control VII*, pp. 235–255, 2001.

[7] A. Cagni, E. Carpaneto, G. Chicco, and R. Napoli, "Characterisation of the aggregated load patterns for extraurban residential customer groups," in *Proc. 12th IEEE Mediterranean Electrotech. Conf. (MELECON '04)*, 2004, vol. 3, pp. 951–954.
[8] E. Carpaneto and G. Chicco, "Probabilistic characterisation of the aggregated residential load patterns," *Generation, Transmission, Distribution, IET*, vol. 2, no. 3, pp. 373–382, 2008.
[9] F. Chen, J. Dai, B. Wang, S. Sahu, M. Naphade, and C.-T. Lu, "Activity analysis based on low sample rate smart meters," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discov., Data Mining*, 2011, pp. 240–248.
[10] J. Chen, W. Li, A. Lau, J. Cao, and K. Wang, "Automated load curve data cleansing in power systems," *IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 213–221, Sep. 2010.
[11] S.-Y. Chen, S.-F. Song, L. Li, and J. Shen, "Survey on smart grid technology," *Power Syst. Technol.*, vol. 33, no. 8, pp. 1–7, 2009.
[12] H. David, "Robust estimation in the presence of outliers," *Robustness Statist.*, vol. 1, pp. 61–74, 1979.
[13] L. Davies and U. Gather, "The identification of multiple outliers," *J. Amer. Statist. Assoc.*, vol. 88, no. 423, pp. 782–792, 1993.
[14] H. Farhangi, "The path of the smart grid," *IEEE Power Energy Mag.*, vol. 8, no. 1, pp. 18–28, Jan.-Feb. 2010.
[15] T. S. Ferguson, "On the rejection of outliers," in *Proc. 4th Berkeley Symp. Math. Statist. Probab.*, 1961, vol. 1, pp. 253–287.
[16] A. J. Fox, "Outliers in time series," *J. R. Statist. Soc. Ser. B (Methodol.)*, pp. 350–363, 1972.
[17] U. Gather, "Testing for multisource contamination in location/scale families," *Commun. Statist.-Theory Meth.*, vol. 18, no. 1, pp. 1–34, 1989.
[18] D. Hammerstrom, R. Ambrosio, J. Brous, T. Carlon, D. Chassin, J. DeSteese, R. Guttromson, G. Horst, O. Järvegren, and R. Kajfasz *et al.*, "Pacific northwest gridwise testbed demonstration projects," *Part I. Olympic Peninsula Project*, 2007.
[19] P. J. Huber, *Robust Statistics*. New York, NY, USA: Springer, 2011.
[20] H. Imai and T. Asano, "Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane," *J. Algorithms*, vol. 4, no. 4, pp. 310–323, 1983.
[21] E. M. Knox and R. T. Ng, "Algorithms for mining distance-based outliers in large datasets," in *Proc. Int. Conf. Very Large Data Bases*, 1998.
[22] H.-P. Kriegel and M. Pfeifle, "Density-based clustering of uncertain data," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery in Data Mining*, 2005, pp. 672–677.
[23] G. M. Ljung, "On outlier detection in time series," *J. R. Statist. Soc. Ser. B (Methodol.)*, pp. 559–567, 1993.
[24] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *J. ACM*, vol. 41, no. 5, pp. 960–981, 1994.
[25] G. Mateos and G. B. Giannakis, "Robust nonparametric regression via sparsity control with application to load curve data cleansing," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1571–1584, Apr. 2012.
[26] A. Nairac, N. Townsend, R. Carr, S. King, P. Cowley, and L. Tarassenko, "A system for the analysis of jet engine vibration data," *Integrated Comput.-Aided Eng.*, vol. 6, no. 1, pp. 53–66, 1999.
[27] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," *ACM SIGMOD Rec.*, vol. 29, no. 2, pp. 427–438, 2000.
[28] W. Schmid, "The multiple outlier problem in time series analysis," *Australian J. Statist.*, vol. 28, no. 3, pp. 400–413, 1986.
[29] R. P. Singh, P. X. Gao, and D. J. Lizotte, "On hourly home peak load prediction," in *Proc. IEEE 3rd Int. Conf. Smart Grid Commun. (SmartGridComm)*, 2012, pp. 163–168.
[30] J. W. Tukey, *Exploratory Data Analysis*. Reading, MA, USA: Pearson, 1977, p. 231.
[31] "Wikipedia," Determining the Number of Clusters in a Data Set [Online]. Available: http://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set Accessed: 2013-11-21

Guoming Tang received the B.S. and M.S. degrees from the National University of Defense Technology (NUDT), Changsha, China. He is currently pursuing the Ph.D. degree in the Department of Computer Science, University of Victoria, BC, Canada.

His research interests include data cleansing and data mining in smart grid.

Kui Wu (S'98–M'02–SM'07) received the B.Sc. and the M.Sc. degrees in computer science from Wuhan University, Wuhan, China, in 1990 and 1993, respectively, and the Ph.D. degree in computing science from the University of Alberta, Alberta, BC, Canada, in 2002.

He joined the Department of Computer Science, University of Victoria, Victoria, BC, Canada, in 2002 and is currently a Professor there. He is also currently an Adjunct Professor in Shanghai University of Electric Power. His research interests include smart grid, mobile and wireless networks, and network performance evaluation.

Jingsheng Lei received the Ph.D. in computer science from Xinjiang University, Ürümqi, China.

He is currently a Professor and the Dean of the School of Computer and Information Engineering, Shanghai University of Electric Power, Shanghai, China. His research interests include data mining, machine learning, and smart grid.

Zhongqin Bi received the Ph.D. degree from East China Normal University, Shanghai, China, in 2009.

He is currently an Associate Professor in the School of Computer and Information Engineering, Shanghai University of Electric Power, Shanghai, China. His research interests include smart grid, data quality control, and cloud computing.

Jiuyang Tang received the Ph.D. degree from the National University of Defense Technology (NUDT), Changsha, China.

He is currently an Associate Professor at the Institute of Information System and Management, NUDT. His current research interests include big data analysis, P2P, and ubiquitous networks.