Manuscript Details

Manuscript number	FGCS_2018_1518
Title	Two Approaches for Synthesising Scalable Residential Energy Consumption Data
Article type	Full Length Article

Abstract

Research on integrated systems that uses simulations to develop demand-side management algorithms for energy use in the building sector requires scalable and detailed energy consumption data. However, due to privacy issues, it is often difficult to obtain sufficiently large data sets. This paper proposes two different methods for synthesizing fine-grained energy consumption data for residential households, namely a regression-based method and a probability-based method. They each use a supervised machine learning method, which trains the models with a real-world data set and then generates large-scale time series based on the models. This paper describes the data generation process, the optimization techniques, and the parallel data generation for a building cluster. This paper evaluates the two time-series generators and compares the resulting consumption profiles with real-world data in detail, including patterns, statistical information, and data generation performance in the cluster. The results demonstrate the effectiveness of the proposed methods and their efficiency in generating large-scale data sets.

Keywords	Energy Consumption; Time series; Synthesise; Simulation; Data generation

Corresponding Author	Xiufeng Liu
Corresponding Author's Institution	Technical University of Denmark
Order of Authors	Xiufeng Liu, Nadeem Iftikhar, Huan Huo, Rongling Li, Per Sieverts Nielsen
Suggested reviewers	Guandong Xu, Omid Ardakanian, YIFENG ZENG

Submission Files Included in this PDF

File Name [File Type]

Cover letter.pdf [Response to Reviewers]

hightlight.pdf [Highlights]

SmartMeterDataSimulator.pdf [Manuscript File]

author.pdf [Author Biography]

picture.pdf [Author Photo]

To view all the submission files, including those not included in the PDF, click on the manuscript title on your EVISE Homepage, then click 'Download zip file'.

Two Approaches for Synthesising Scalable Residential Energy Consumption Data

Xiufeng Liu[a,*], Nadeem Iftikhar[b,*], Huan Huo[c], Rongling Li[a], Per Sieverts Nielsen[a]

[a] *Technical University of Denmark*
[b] *University College of Northern Denmark*
[c] *University of Technology Sydney, Australia*

Abstract

Research on integrated systems that uses simulations to develop demand-side management algorithms for energy use in the building sector requires scalable and detailed energy consumption data. However, due to privacy issues, it is often difficult to obtain sufficiently large data sets. This paper proposes two different methods for synthesizing fine-grained energy consumption data for residential households, namely a *regression-based* method and a *probability-based* method. They each use a supervised machine learning method, which trains the models with a real-world data set and then generates large-scale time series based on the models. This paper describes the data generation process, the optimization techniques, and the parallel data generation for a building cluster. This paper evaluates the two time-series generators and compares the resulting consumption profiles with real-world data in detail, including patterns, statistical information, and data generation performance in the cluster. The results demonstrate the effectiveness of the proposed methods and their efficiency in generating large-scale data sets.

Keywords: Energy Consumption, Time series, Synthesise, Simulation, Data generation

1. Introduction

Today, smart meters are being widely installed to collect energy consumption data in the building sector. At the same time, utilities are showing an increasing interest in building energy data management systems in order to improve their services and decision-making processes [42]. The ability to handle big data sets has become a mandatory requirement for todays energy data management systems. On the one hand, when building such systems, large data sets are needed for investigating suitable technologies and algorithms. On the other hand, scalable data sets are required for benchmarking the systems before deployment to production, such as evaluating system performance, robustness, and scalability. Big real-world energy consumption data sets, such as smart meter data, can meet these goals, but the challenge is that it is often difficult to obtain scalable real-world data sets, mainly due to privacy issues. This is because energy consumption data usually contains sensitive information. For example, the living

*Corresponding author
Email addresses: xiuli@dtu.dk (Xiufeng Liu), naif@ucn.dk (Nadeem Iftikhar)

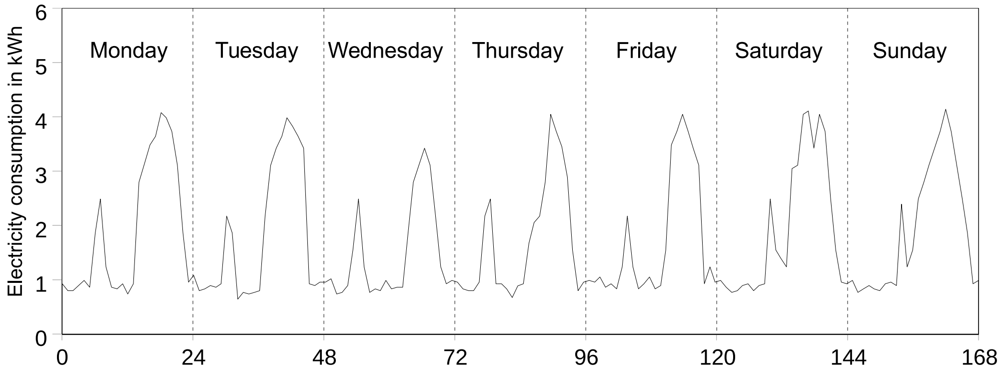

Figure 1: Weekly consumption pattern of a typical household

habits of a household can be revealed through consumption pattern analysis. To date, there are only a few open energy consumption data sets available. These data sets are anonymized but they are limited in size, such as [23, 33, 43, 7, 49]. Many countries have restricted the dissemination and use of personally relevant data by law, including the Scandinavian countries, Denmark and Sweden. The recent enforcement of the EU General Data Protection Regulation (GDPR) [18] mandates strong privacy protection for personal data. This makes it more difficult to publish any data with a bearing on personal privacy, such as energy consumption data. A data generator is therefore required to generate data sets that simulate realistic energy consumption.

Scalable smart meter data sets have been used in benchmarking studies of time-series management systems and the design of analytic algorithms [30, 31]. Many simulations in the fields of energy, climate, and buildings require residential electricity consumption profiles. A typical example is research on building occupancy based on fine-grained energy consumption data. Load profiles vary between households, due, for example, to family size, age, and living habits. Using an average load profile may lead to inaccurate results, even to errors. Synthetic data is usually the only choice due to the lack of measured load profiles.

Synthetic data generation is an effective way of developing electricity consumption time-series data. However, it is often a non-trivial task to simulate real-world consumption data, due to the difficulty of reproducing time series characteristics, including trend, seasonality, and pattern. Figure 1 shows an example of a fragment of an electricity consumption time series for one week. It has a regular daily pattern, with a peak in the morning and another in the evening. The morning peak appears earlier on a weekday than at the weekend, as the household gets up earlier for work. The second peak of the weekend lasts longer than on a weekday, which may be because the family spends more time at home during the weekend, and uses more energy. A synthetic time series should be able to reflect this information.

In this paper, we present two distinct approaches for generating scalable realistic energy consumption data sets, one using a regression-based method and the other using a probability-based method. These approaches are both supervised machine-learning methods, including a training process and a data generation process. The regression-based method allows different auxiliary data for the input, in addition to the seeded time series, and these include indoor activities, outdoor temperature time series, appliance parameters, building data and other related data. In the absence of these data, we used the simplest auto-regressive method to synthesise new consumption values by prediction. In addition, we took a number of steps to optimize the data generator in order to reproduce real-world consumption time series characteristics as well as possible, including de- and re-seasonalisation, clustering, adding base load and white noise. In contrast, the probability-based method requires only the seeded data as input. This approach first identifies representative consumption patterns by clustering, then establishes a probability model, and generates new time series by a random walk on a Markov chain. In order to optimize large-scale data generation, we implemented the proposed methods using a memory-based distributed computing framework, *Spark*. This paper is based on a conference paper [22], but with a significant extension: we have added the probability-based generation method, comprehensively compared the proposed two methods, and discussed several related issues regarding their differences and their selection for different cases.

The main contributions of this paper are as follows:

- We propose two distinctly different and novel approaches for fine-grained smart-meter data generation.
- We investigate how to simulate real-world energy consumption time series more effectively, including the preservation of patterns, seasonality, and segmentation groups.
- We propose and implement two data generators that can generate large-scale data sets in parallel in a cluster.
- We comprehensively evaluate these data generation methods, evaluate their effectiveness in simulating real-world data, the scalability of generating large data sets, and their comparison.

The paper is organized as follows. Section 2 reviews related work. Section 3 describes the two data generation methods. Section 4 describes the parallel data generation implemented on Spark. Section 5 evaluates the two types of generators. Section 6 presents conclusions and suggestions for future work.

2. Related Work

The synthesis of energy consumption time series data sets has been a topic of increasing interest in recent decades. The first approach [3] was to build a load profile based on a simple probability model of appliance use, for example, by assuming a 90% probability that TV sets would be on during the time 19:00–21:00. This approach can approximate the energy consumption of individual appliances, but the combined effects for many appliances, used to obtain the overall consumption profile, are distorted. A recent study [41] was based on a further development of this idea, and generated more accurate realistic load profiles. The approach was to match appliance use to indoor activities, using them to generate consumption profiles based on a statistical model. This approach can achieve better accuracy but requires additional data on household activities, which are often not easy to obtain. Other approaches have used a bottom-up approach to simulate the electricity consumption of a household from the use of home appliances [34, 40, 44, 2, 15]. The drawback is that this approach requires detailed consumption data for each type of home appliance, which is also difficult to obtain. For this reason, most research has used the statistical average of electrical consumption instead, as in [16, 1]. In contrast, the two approaches proposed in the present paper both exploit machine learning methods that use a small set of real-world consumption data as the seed to generate a consumption data set, although one is regression-based and the other is probability-based. These new approaches make direct use of the seeded data to simulate its time-series characteristics, including pattern, seasonality, and variability, and are thus more efficient and realistic.

The research that is relevant to the data generation models used in the present paper are as follows. The proposed regression-based method uses an autoregressive centred moving average model, which is an improved version of an autoregressive moving average (ARMA) [21]. There are many other time series forecast models, including autoregressive integrated moving average (ARIMA) and neural networks, as proposed in [47]. Ref [4] generated time series using the periodic autoregressive moving average model (PARMA) that takes into account the seasonality (or period) of a time series. Additionally, ref [32] used periodic autoregressive model with exogenous variables (PARX) for short-term energy consumption prediction. This model combines exogenous variables, such as building area, ages, types, outdoor temperatures, and resident characteristics for further improving prediction accuracy. Ref [11] conducted a survey of time series forecasting, and concluded that stochastic, neural networks, ARIMA and its variants now play a major role in time series forecasting. However, it is worth noting that other machine learning methods, Support Vector Machines (SVMs) and Artificial Neural Networks (ANNs), are also widely used for energy consumption prediction, e.g., [14, 27, 13]. In recent years, as Deep Learning (DL) methods [8] have been developed, they are also being used for energy prediction, as in [35, 17], which predict new values through the characterisation of demand profiles based on measured data. DL uses multiple layer computational models to learn representations of data, resulting in a more powerful prediction capability than is obtained by artificial neural networks [26].

The research that is the most closely related to the proposed probability-based approach is [5], which used a Markov chain model to generate an Internet-of-Things (IoT) time series. However, their approach creates a transition probability matrix (TPM) for each step on the time axis, which results in a large TMP matrix and greatly increases the computational cost. In contrast, we build the TPMs based on the representative daily patterns of energy consumption series, which can achieve a smaller TMP and more computing efficiency.

As an emerging technology, smart metering has received much attention recently, leading to developments in the field of smart energy. However, to the best of our knowledge, very little research on smart meter data simulation or generation has been carried out. Ref [39] generated residential load profiles based on pre-defined household templates, while [30] used a PARX model to generate time series for system benchmarking purpose. In contrast to these works, our regression-based approach has further optimized a simulation that uses clustering technique to preserve customer segmentation information and implemented parallel consumption time series generation for a cluster.

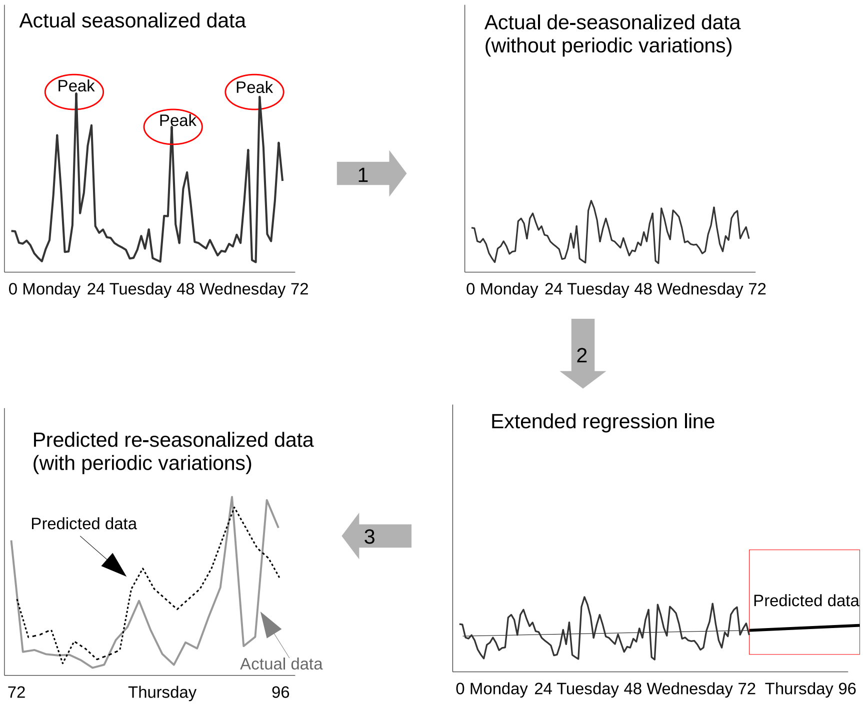

Figure 2: Overview of the regression-based data generation method

3. Methods

This section describes the regression-based and the probability-based data generation methods.

3.1. *The regression-based method*

The regression-based method uses prediction to generate new time series values. It is a supervised machine learning method consisting of a training process and a data generation process. A residential energy consumption time series has a number of characteristics, including *trend*, *cyclicity* and *seasonality/periodicity*, so to improve the simulation of real-world data, some pre- and post-processing of the data is required. Figure 2 shows an overview of the regression-based method. In the preprocessing, we first flatten the periodic variation for the training data, which is called *de-seasonalisation*. The reason is that a model trained on a de-seasonalised time series can achieve better prediction accuracy [48]. We then use this model to generate new predicted consumption values. In post-processing, the periodic variations re-applied to the new time series, with a base load representing a fixed consumption for the day. This process is called *re-seasonalisation*, which is the last step in the data generation.

In the following, we provide more details on the regression-based data generation method, including algorithms and optimisation techniques.

3.1.1. Model training process

For a time series, $X =< x_1, x_2, ..., x_n >$, the training process consists of three sequential steps, including fluctuation flattening of the time series, de-seasonalisation and training the autoregressive model. The training output is used by the generation process to synthesise a consumption time series data set. The three steps are as follows:

Fluctuation flattening. The *Centred Moving Average (CMA) method* [45] is used to flatten the periodic fluctuations of a time series. CMA is the sliding-window approach that uses the mean value of the time series values within the window to replace the original value for each interval. In the present paper, we used a daily load profile with a window size of 24 hours and a sliding interval of 1 hour. The i-th value of a CMA flattened time series, $C(i)$, can be defined by Equation 1. Note that as the period of 24 hours is an even number, we use the mean of two adjacent values as the CMA value, i.e.:

$$C(i) = \frac{1}{2}\left(\frac{x_{i-12} + .. + x_i + .. + x_{i+11}}{24}\right) + \frac{1}{2}\left(\frac{x_{i-11} + .. + x_i + .. + x_{i+12}}{24}\right) \tag{1}$$

where x_i represents the i-th observation of a time series.

De-seasonalisation. De-seasonalisation is used to reduce the periodic aspect of a time series, as follows. First, the so-called *Ratio-to-Moving-Average* or *Raw-index* is defined as follows:

$$\mathcal{R}(i) = \frac{x_i}{C(i)} \tag{2}$$

Then, we compute the *periodic index* for each hour of the day based on the raw index values (see Equation 3). The periodic index for each hour of the day is the mean value of the raw indices at the same hour in all days. For instance, $\mathcal{I}(0)$ is the mean of the values of $\mathcal{R}$ at 0 o'clock in all days. There is therefore a total of 24 periodic indices, each of which corresponds to one hour of the day.

$$\mathcal{I}(h) = \frac{1}{n}\sum_{i=0}^{n-1} \mathcal{R}(h + 24i) \tag{3}$$

where n is the number of days in a time series, and h is the hour of the day, i.e., 0 – 23. Due to the floating point [25], there exists a precision problem for the value of $\mathcal{I}$. It is therefore necessary to normalise $\mathcal{I}$ to ensure that the sum of the periodic is equal to 1.0. Equation 4 performs the normalisation, and derives the normalised periodic indices, I':

$$\mathcal{I}'(h) = \frac{24 * \mathcal{I}(h)}{\sum_{h=0}^{23} \mathcal{I}(h)} \tag{4}$$

Finally, the time series is de-seasonalised by using a normalised periodic index, yielding a flattened time series, $X' =< x'_1, ..., x'_i, ..., x'_n >$, in which

$$x'_i = \frac{x_i}{\mathcal{I}'(h)} \tag{5}$$

where $x_i \in X$ and $h = i \; mod \; 24$.

Training an autoregressive model. The de-seasonalised time series X' is then used to train the autoregressive (AR) model. As mentioned earlier, models trained on a flattened time series obtain better prediction accuracy. The predicted values will be used to construct the final consumption time series. According to [19], the time series values of residential energy consumption are serially co-related, i.e., current consumption is related to past consumption, as verified by the experiment reported in Section 5.3.2. According to [19, 32, 6], it is appropriate to choose the order of three for the autoregression, i.e., $p = 3$.

In summary, the above training process will result in the following outputs, including periodic indices, flattened time series, and AR models. The results will be passed to the data generation process for synthesising a time series. In the present application, we employed a distributed computing system, *Apache Spark*, for parallel generation. In our implementation, we therefore wrote the output directly to the Hadoop distributed file system (HDFS), and organised the output into two separate text files: one for storing periodic indices and the other for storing the AR models and the flattened time series. The records in the two files are linked by unique IDs. The purpose of this implementation was to generate a large number of realistic time series by combining the records of the two files, as discussed in the next section.

3.1.2. Time-series generation process

We now describe the time series generation process using Algorithm 1. The time-series generation process uses the periodic indices, the autoregressive data and the flattened time series as the input for generating data. As mentioned earlier, we generated scalable time series on the distributed computing platform, Apache Spark. These parameters were saved in a Hadoop distributed file system, and read into Spark as two Resilient Distributed Datasets (RDDs), $\mathcal{PI}$ and $\mathcal{AR}$ in Spark. *Theta join* [37] was then applied to the two RDDs to generate new time-series values (see line 1–6). Theta join is able to generate a time series by combining the parameters from two tables, which makes it possible to generate large-scale data sets with a relatively small seed.

The data generation process was follows: (1) *Generate new consumption values:* A new consumption value is generated based on the following autoregressive function:

$$x_i'' = c + \sum_{\lambda=1}^{p} \alpha_\lambda x_{i-\lambda}' \tag{6}$$

where c is a constant intercept, α_λ are coefficients, and x_i' are the flattened time series values (using p values before i); *(2) Re-seasonalize the time series, and (3) add base load and white noise*, as expressed by Equation 7.

$$x_i''' = x_i'' * \mathcal{P}'(h) + baseLoad + \epsilon_i \tag{7}$$

where $h = i \; mod \; 24$, $i = 1, ..., n$ and ϵ is white noise. The re-seasonalisation is achieved by simply multiplying the flattened time-series value by the corresponding adjusted periodic index. A base load is added to simulate the energy consumption that is independent of the activities of a household, for example, the consumption used by the appliances

Algorithm 1: Energy consumption time series generation

Input : The periodic indices $\mathcal{PI}$(id, peridoc-indices), autoregressive models and flattened time series $\mathcal{AR}$(id, AR-coefs, flattened-time-series)

Output: A set of synthetic time series

$\mathcal{R} \leftarrow \mathcal{PI} \bowtie_{\mathcal{PI}.id \neq \mathcal{AR}.id} \mathcal{AR}$; `/* Theta join */`

$O \leftarrow \{\}$;

for $r \in \mathcal{R}$ **do**

  $x'' \leftarrow$ Generate new values using *r(AR-coefs, flattened-time-series* x'*)* by prediction;

  $x''' \leftarrow$ Re-seasonalize x'' using *r(periodic indices)*, and add baseload and white noise ;

  $O \leftarrow O \oplus x'''$;

end

return O

that are always on, e.g., refrigerators. The base load value can be obtained by averaging the consumption in the middle of the night or the consumption when people are away from home. A more common approach is to use 10% of the average hourly consumption value to represent the base load of a household [9], and in this paper we employ this approach. Finally, we add white noise to simulate a slight variation of each hourly consumption value. The white noise conformed to a standard normal distribution: $\epsilon \sim \mathcal{N}(0, 1.0)$.

3.1.3. Optimising data generation

The optimisation techniques that we applied to this time series data generator will now be described. As discussed in Section 1, residential energy consumption time series have regular time patterns, such as daily, weekly or monthly. In fact, the appearance of these regular patterns is a complex issue, which is related to many factors, such as changes in the weather, building characteristics and living habits. These patterns may also show spatial and temporal characteristics. For example, the behavioural patterns of the residents in the same neighbourhood may be similar, as are patterns within a certain time period. Utilities often use a clustering technique to identify customers with similar patterns in order to provide better energy services or personalised energy-saving recommendations. However, in the generation process, due to the use of theta join, the models and the flattened time series are shuffled to synthesise a time series. This operation will lead to the loss of customer segmentation information from the original time series. In order to preserve segmentation information, we optimise the training process by adding a preprocessing step of clustering (see Figure 3). The remaining training and generating process is then conducted using the clustered seeds separately.

More specifically, we first cluster the seed based on representative daily consumption patterns of all time series. A representative consumption pattern is the mean pattern of a time series calculated by averaging the consumption values at the same hour for all days. For example, for a time series of i, its representative daily pattern can be defined

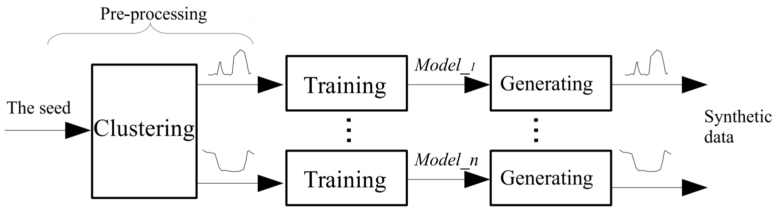

Figure 3: Preprocess the seed by clustering

as

$$\bar{X}_i = \{\bar{x}_{i,0}, \bar{x}_{i,1}, .., \bar{x}_{i,23}\} \tag{8}$$

where $\bar{x}$ is the mean of the consumption values at that hour of the day, i.e., 0 – 23.

We then apply k-means clustering algorithm [46] to cluster all the representative daily patterns. Usually, k-means clustering uses the Euclidean distance as the metric to quantify the similarity between two vectors, e.g., [38, 28]. In our case, the distance of two representative daily load profiles, $\bar{X}_i$ and $\bar{X}_j$, can be computed by the following equation:

$$euclDist\left(\bar{X}_i, \bar{X}_j\right) = \sqrt{\sum_{h=0}^{23}\left(\bar{x}_{i,h} - \bar{x}_{j,h}\right)^2} \tag{9}$$

However, in the present study we chose the Pearson correlation distance [10] for the optimisation of clustering. The correlation metric is defined as follows:

$$corrDist\left(\bar{X}_i, \bar{X}_j\right) = 1 - corr\left(\bar{X}_i, \bar{X}_j\right) \tag{10}$$

where:

$$corr\left(\bar{X}_i, \bar{X}_j\right) = \frac{\sum_{h=0}^{23}\left(\bar{x}_{i,h} - \mu_i\right)\left(\bar{x}_{j,h} - \mu_j\right)}{\sqrt{\sum_{h=0}^{23}\left(\bar{x}_{i,h} - \mu_i\right)^2}\sqrt{\sum_{h=0}^{23}\left(\bar{x}_{j,h} - \mu_j\right)^2}} \tag{11}$$

where μ is the mean of the representative daily patterns.

The reason is that the correlation distance is better for measuring the shape or trend of two patterns, while the Euclidean distance is for measuring the difference of attributes in values [20]. For example, the Euclidean distance of Figure 4 (a) and (b) are both $\sqrt{3}$, but we can see that the two patterns in Figure 4 (b) are completely different.

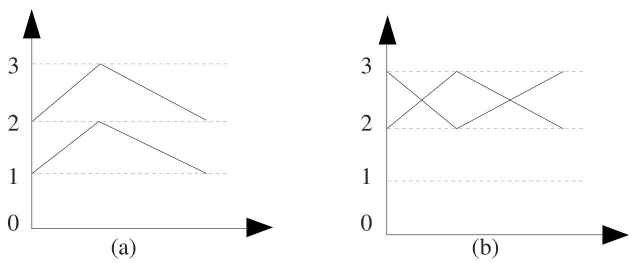

Figure 4: Two identical or opposite patterns

As the correlation value ranges between -1 and 1, the correlation distance will be between 0 and 2. Usually, a distance of less than 0.5 represents a good similarity between two vectors. If the distance of two vectors is 0, they have identical patterns, as in Figure 4 (a). If the distance is 2, they have opposite patterns, as in Figure 4 (b).

3.2. *The probability-based method*

The alternative approach using a probability-based method to simulate an energy consumption time series will now be described. It is a two-step method, which includes representative pattern extraction by clustering and new time series generation using a probability model. The probability model is constructed using representative daily patterns (see Figure 5). The procedure is described in the following.

3.2.1. *Extracting representative patterns*

We use the adaptive clustering method [24] to extract representative patterns for each customer. We first normalise the daily load profiles for each household. The normalisation process is defined as follows. For a household, i, the load profile of the d-th day can be represented by $X_i(d)$, where X represents an hourly consumption vector with 24 dimensions, $X \in \mathbf{R^{24}}$ and $d = 1, ..., N$. The normalised daily load profile is defined as follows:

$$X_i^*(d) = \frac{X_i(d)}{S_i(d)} \tag{12}$$

where X^* is the normalised daily load profile and $S_i(d)$ is the total energy consumption of day d. Then, the adaptive clustering is conducted based on all normalised load profiles, and the centroids of the clusters will be used as the representative load profiles of a household. As the clustering is based on the normalised data, the representative patterns derived indicate only the shapes of the consumption pattern, without indicating consumption intensity. The shapes can often reflect the consumption habits or activities of a household. Finally, the representative patterns are encoded using ASCII alphabets.

3.2.2. *Time series generation*

The time series generation includes the following procedures. First, based on the derived representative patterns, each time series is converted into a sequences of ASCII alphabets. Then, the sequences are used to train a Markov chain model, which is then used to generate new sequences by random walks. The new sequence represents the normalised consumption patterns within a series of continuous days. The sequence is then "amplified" to create a consumption time series by multiplying a random number sampled from the daily consumption distribution generated by the training data set.

Markov chains are often used for sequence generation, e.g. [5, 36, 29]. A discrete-time Markov chain can be defined as a finite set of states, $S = 1, ..., n$, representing the events that occur at every discrete time step. The next state in a Markov chain is conditionally independent of the past states, i.e.:

$$P(S_{t+1} = j | S_t = i, S_{t-1} = i_{t-1}, ..., S_0 = t_0) = P(S_{t+1} = j | S_t = i) \tag{13}$$

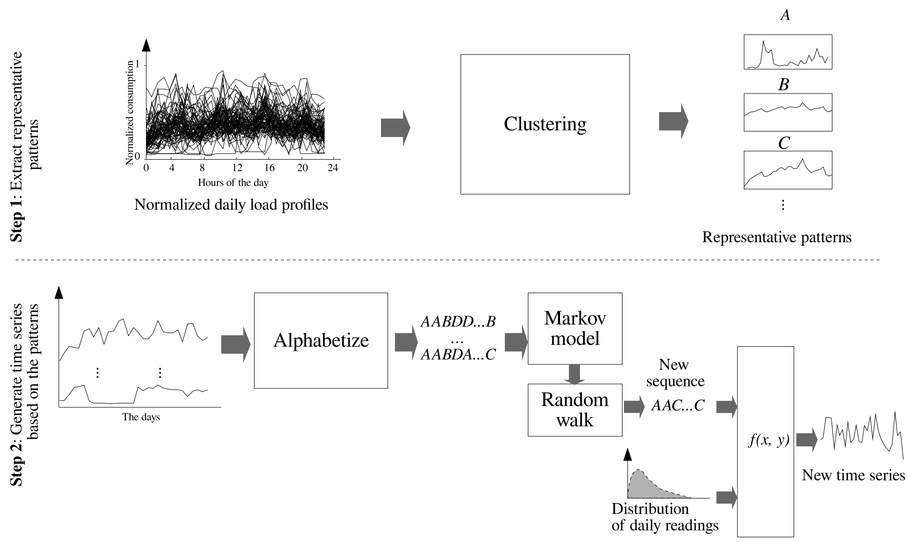

Figure 5: Overview of the probability-based simulation method

where $P(S_{t+1} = j|S_t = i)$ is the probability of the transition between two states i, j. A transition probability matrix (TPM) of the size $n \times n$ is created for each concrete time step. TPM contains all the probabilities of the state transitions.

We compute the transition probability P_{ij} by the following equation:

$$P_{ij} = \frac{n_{ij}}{\sum_{k \in S} n_{ik}} \tag{14}$$

where the numerator represents the number of daily pattern changes from i to j between two continuous days, and the denominator represents the number of pattern changes from i to all states (including state i), $\sum_{k \in S} P_{ik} = 1$. When constructing the TPM, it should be noted that the training data sets may not include the transitions between all states. This will generate a sparse TPM. To address this issue, Laplace smoothing is used to increase the transitions by adding the number 1 such that there will be no zero probability in the resulting TPM.

When the TPM for each time step has been derived, we generate new sequences by random walks on the Markov chain. We start from a randomly selected state, then pick each subsequent state according to the TPM corresponding to that particular time step. The resulting alphabet sequence represents a series of synthetic normalised consumption patterns. When a new alphabet sequence has been generated by a random walk, the load profiles of all days are determined, each of which is generated by the normalised daily pattern multiplied by a random number sampled from the corresponding daily consumption distribution. To capture the seasonality of the consumption behaviours, we create a distribution for each day using the daily consumption values of all households in order that the sampled random number can reflect the consumption change over time. For example, in countries with widespread use of electric heating, the daily consumption in winter is typically higher than it is in summer, as in Ireland and Canada.

4. Parallel data generation

We used Apache Spark for parallel data generation. Apache Spark is an open source memory based distributed computing framework, which has implemented the distributed computing primitives, including *map* and *reduce*. Spark is optimised for iterative algorithms and interactive data analysis, which can perform iterative computations on the same data set. A Spark cluster has the architecture of one master node and multiple slave nodes. The master node assigns jobs to the slave nodes and coordinates the jobs run in a cluster. A job reads the data from a Hadoop distributed file system (HDFS) or in a local machine and performs computations on a resilient distributed data set (RDD), an in-memory data structure partitioned across the nodes that operate in parallel. The output is written to HDFS or a local machine. A job can be composed of several steps that are either maps or reduces. All the data is split into multiple partitions and the computations are performed on each partition by a separate task. A task is executed by an executor on each slave node.

Using the computation mechanism of Spark, we implemented parallel data generation for the proposed two methods, for both the training and the generation programs. It is worth noting that the training program does not have to be implemented using Spark as it is run only once, and the resulting models can be re-used many times to generate data. In the following we therefore describe only how to parallelise data generation on Spark. For both of the methods, a map-only data generation program was implemented, i.e., no reducer was needed. The models generated by the training process were broadcast to the mappers which generated time series separately without inter-communication. This greatly improves performance when generating large-scale data sets (this will be evaluated by the experiments). For the regression-based method, the broadcast data are periodic indices, auto-regression coefficients, and base loads. In a mapper, a theta join is first run to generate a new time series using the PI and AR parameters, then the time series is re-seasonalised to simulate the change of consumption over time. Base load and white noise were then added. For the Probability-based method, the broadcast data included the transition probability metrics and distribution models of daily readings. Time series generation was conducted by random walks within a mapper. In both methods, the generated time series were then written directly as the map output to HDFS.

5. Evaluation

This section reports an evaluation of the two proposed time series generation methods. An Irish electricity consumption data set [23] was used for training the models. This data set was released by the Commission for Energy Regulation in Ireland and was recorded from July 14, 2009 to December 31, 2010 with over 5,000 residential households and businesses. The original data set has a 30-minute resolution. We merged them into an hourly resolution for the experiments and in the present paper we consider only residential consumption. Both of the data generation methods used clustering analysis in the training process. There is no rule-of-thumb about the least sample size for clustering analysis [12]. As we intend to maintain a relatively small size of the seed for generating data, we randomly sampled 30% of the time series as the seed (the training data).

We evaluated the simulation data by descriptive and exploratory analysis and compared them with the real-world data. The two proposed data generation methods were compared to different settings.

All the experiments were conducted in a cluster of four nodes. All of them used slave nodes, and one of them was used as master node. All the nodes had an identical configuration: Intel CPU E5-2650 (3.40GHz, 4 Cores) with hyper-threading enabled (2 hyper-threads per core), 8 GB RAM, Hard driver (1TB, 6 GB/s, 32 MB Cache and 7200 RPM), and 64bit-Ubuntu 12.04.

5.1. Regression-based results

As described in Section 3.1.3, we preprocessed the seed by clustering before using it to train the models. Also, we used the Pearson correlation distance metrics in the clustering. In the following, we would like to further explain this process by an example before evaluating the time series thus generated.

This example is shown in Figure 6, which includes typical daily load profiles from four households, denoted by TS_{1-4}. According to the energy consumption intensity, TS_3 is the highest, TS_1 is medium, TS_2 and TS_4 are the lowest. According to the pattern, TS_1, TS_2 and TS_3 are similar, as they have a morning peak and an evening peak within an almost identical time window. In contrast, TS_4 has a different pattern, as it has no morning peak and a low consumption at roughly 5 o'clock in the afternoon. Based on pattern similarity, TS_1, TS_2 and TS_3 should therefore be in the same group, while TS_4 should be in another group.

We now compare the Euclidean distance and the correlation distance for clustering time series according to pattern similarity. Table 1 shows the pair-wise comparison of the distances for the daily load patterns in Figure 6. According to the correlation distances, the distances of the pairs, (TS_1, TS_2) and (TS_1, TS_3), both are smaller than (TS_1, TS_4). In contrast, the Euclidean distance of the pair, (TS_2, TS_4) is the smallest, so if the Euclidean distance metric were used, the load profiles of TS_2 and TS_4 would be assigned to the same group. This demonstrates that it is more preferable to use the correlation distance metric for clustering consumption patterns or load shapes.

We will now demonstrate the necessity of preserving customer segmentation information by using the clustering technique. We generated time series by using the models that were trained by the seed with and without preprocessing, respectively. We performed adaptive clustering on the corresponding daily load profiles, and generated 20 clusters. The top three clusters are shown in Figure 7. According to the figure, the load profiles in Figure 7 (a) (using a reprocessed seed) are more cohesive than in Figure 7 (b) (using an un-reprocessed seed). This demonstrates the effectiveness of the proposed clustering technique for achieving pattern preservation.

We compared the synthetic time series with the real-world time series (see Figure 8). The blue line in Figure 8 (a) is the daily load profile of a typical household, while the other two are the synthetic load profiles, which resulted from the preprocessed seed that used the correlation distance (corrDist) and the Euclidean distance (euclDist) metrics for clustering, respectively. In Figure 8 (a), it may be seen that the daily load pattern of synthetic data (*corrDist*) matches well with the real-world load pattern: they both have peaks at the hour of 68 and 1618 (with a slight drift to the left). In contrast, the pattern of the synthetic data (*euclDist*) does not match well with the real-world pattern as the latter

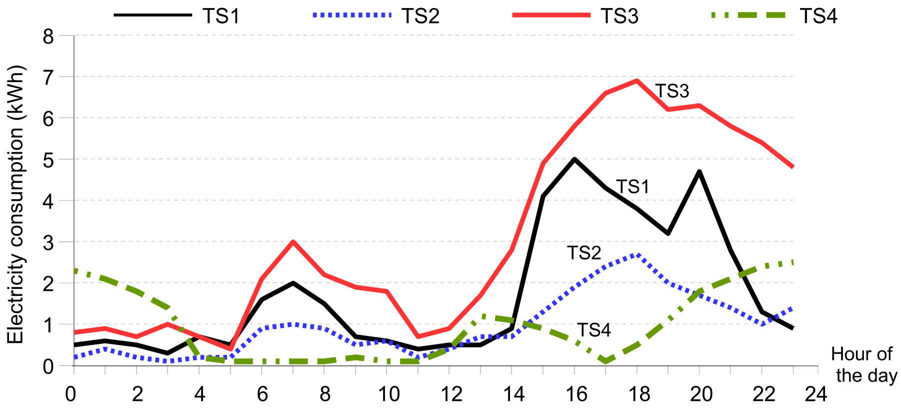

Figure 6: Four typical daily load profiles

Table 1: Pair-wise comparison of the distance metrics

	(TS_1, TS_2)	(TS_1, TS_3)	(TS_1, TS_4)	(TS_2, TS_3)	(TS_2, TS_4)	(TS_3, TS_4)
euclDist	6.13	9.12	9.64	11.5	4.73	12.4
corrDist	0.12	0.13	1.06	0.12	0.76	1.10

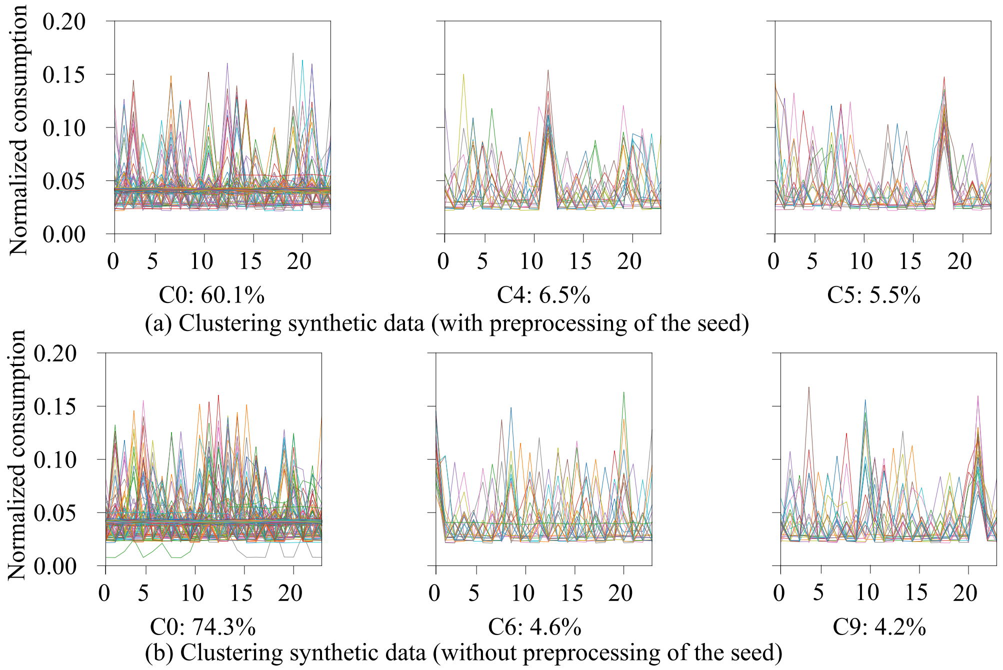

Figure 7: Pattern preservation comparison when the seed was preprocessed and not preprocessed

has no peak at 1–2 o'clock. Figure 8 (b) are the averaging weekly patterns. Here it may also be seen that the pattern of synthetic (*corrDist*) matches better than the synthetic (*euclDist*).

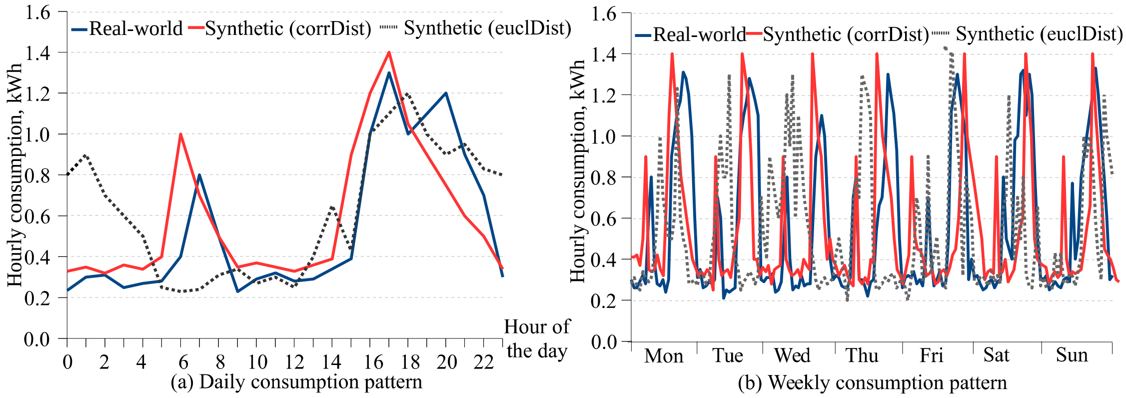

Figure 8: Comparison of consumption patterns

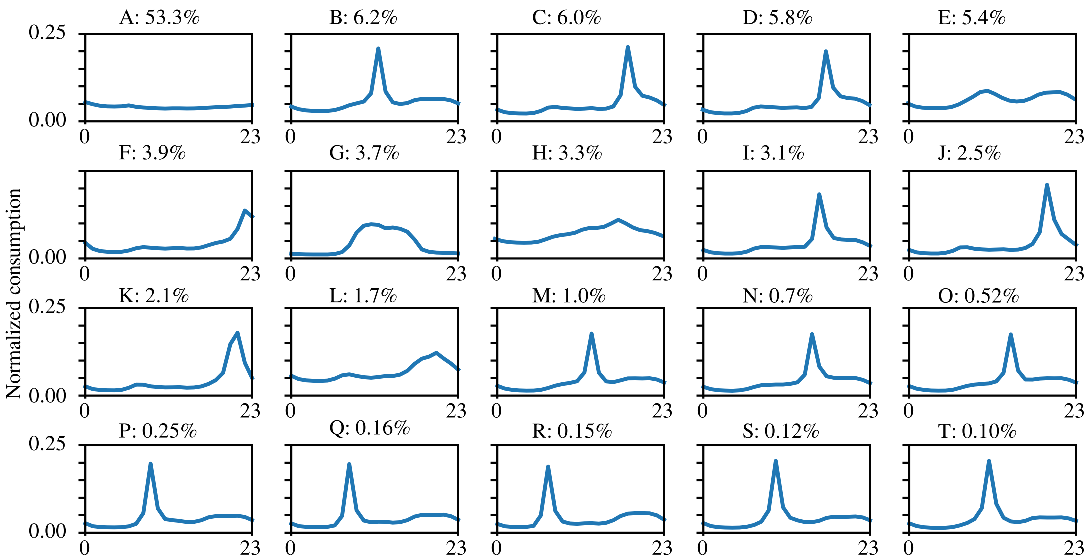

Figure 9: The identified 20 representative patterns

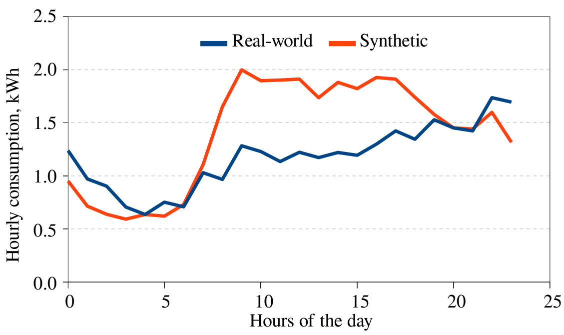

Figure 10: Average daily consumption of June

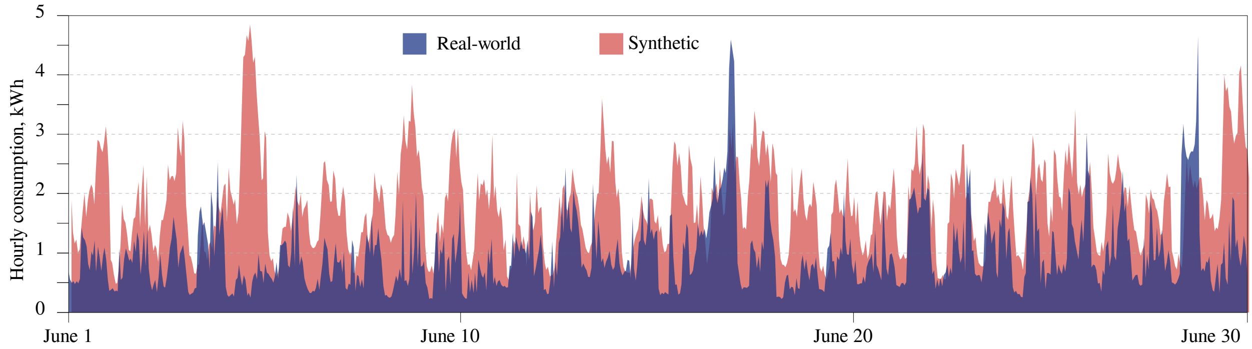

Figure 11: Comparison of consumption time series of June

5.2. *Probability-based results*

For the probability-based method, the representative daily consumption patterns from the training data set must first be identified. We used the seed for training, then used the seed again to validate the results. Adaptive clustering was implemented on the normalised daily load profiles of the seed, which resulted in 20 clusters. Figure 9 shows the clustering results ordered by the number of patterns in the clusters. The 20 representative patterns were labelled with the alphabetic characters from A to T. Each time series was then transformed into a sequence of alphabetic characters according to its daily patterns. Based on the sequences, the TPMs of the Markov chain were calculated for all days, for each of which the probabilities were computed based on Equation 14. A resulting alphabet sequence was converted into a synthetic time series by multiplying the random numbers that were sampled from the corresponding consumption distributions for the days.

We then evaluated the data generator by comparing the real-world and the synthetic data by examining their statistical properties. We took the daily consumption of June as an example and computed the average consumption of each hour of the day (see Figure 10). As may be seen, the shapes of the real and synthetic consumption curves are relatively similar, with low consumption in the early morning, becoming higher during the day and the evening. The consumption profiles of the whole month (June) are shown in Figure 11, which reveals the day-to-day patterns and the discrepancies between the real-world and synthetic consumption time series. The result indicates that the synthetic consumption time series share a very similar pattern with the real-world time series.

Based on the above results, we believe that the probability-based method can generate reasonably realistic consumption data and can therefore be used to evaluate building performance or the consumption behaviour of residents in terms of energy consumption patterns.

5.3. *Comparison*

In the previous subsection, we compared the visual patterns or shapes of the synthetic time series with the real-world data. In the following, we will study the statistical parameters of the data and the possibility of generating large-scale data sets using the proposed methods.

5.3.1. *Statistical performance*

Figure 12 shows the probability distributions of the real-world and synthetic hourly consumption during one month (June). The results indicate that the distribution of the data generated by the regression-based method is more similar to real-world data than data generated by the probability-based method. This is because the latter was generated by the normalised hourly reading the data independently sampled from the consumption distribution of each day. For further investigation, we plotted the distribution of the daily consumption of the real-world data for one of the days in Figure 13. As shown in the figure, the data conformes closely to a standard distribution.

Figure 14 shows a quantitative comparison of the real-world and synthetic data using box-plot method. The box-plot shows the summary statistics including minimum, first quartile, median, third quartile, and maximum, with the outliers shown outside the upper limit. The box-plot shows the statistical parameters for the twelve months of a year and indicates that the three data sets are quite similar. The biggest difference is in the length of the box. The synthetic data generated by the probability-based method is always slightly higher than the real-world data and the synthetic data generated by the regression-based method. This means that the synthetic consumption generated by the probability-based method is more distributed over the month. This may be because we used the Laplace method to smooth a zero probability of the transition between two states in the TPM. This diversifies the transition of patterns in a sequence. On the other hand, the difference may originate from the distribution of the daily consumption, from which we sampled the random number.

Figure 15 shows the auto-correlation of the real-world and the two synthetic data sets, with a time lag of up to 50 hours. Auto-correlation was computed based on the normalised patterns, which is a good way to study the appearance period of repeated patterns. According to the auto-correlation function (ACF) values, the regression-based method provides a better match with the real-world data. It should be remembered that in the regression-based approach, we optimised the model training by using the Pearson correlation distance to improve pattern matching accuracy, and this experiment verifies that this optimisation can yield better results (see also Figure 6 and Table 1). In contrast, the probability-based method generated the pattern sequence by relying on the TPM, and exhibits sub-optimal matching performance in terms of ACF.

5.3.2. *System performance*

System performance including the training and data generation processes will now be examined. It may be recalled that the training models can be re-used in the data generation process. For each method, the training process includes a number of steps, which are shown in Figure 16 (from bottom to top). This figure also shows the corresponding time used in each step when using the full set of the training data. In the regression-based method, the normalisation and K-means clustering are optional steps for the optimisation purpose, indicated by the dashed-line rectangle. As shown in this figure, clustering is the most time-consuming action in both methods as they use adaptive clustering. It is a two-stage method, which first performs adaptive K-means clustering to find the optimal number of clusters using an elbow method, then performs hierarchy-clustering to merge small clusters [24]. The adaptive clustering typically uses

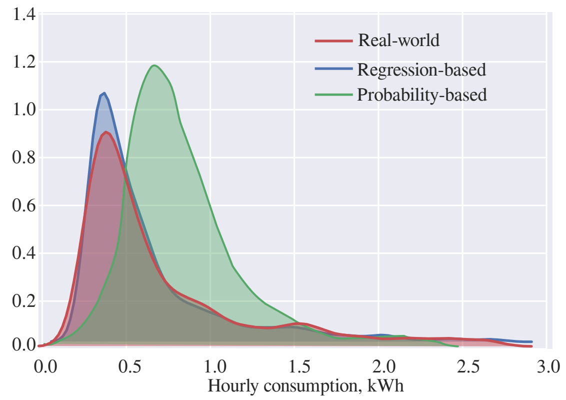

Figure 12: Comparison of hourly reading distribution

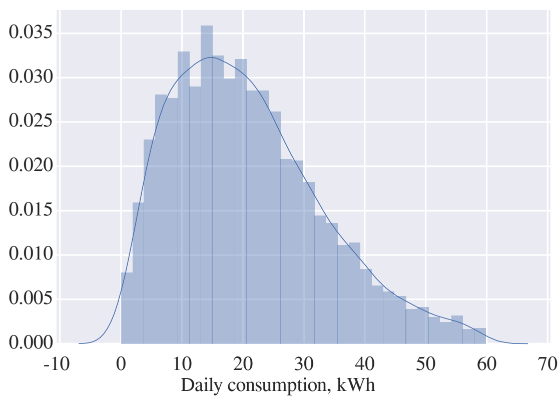

Figure 13: Distribution of daily consumption distribution

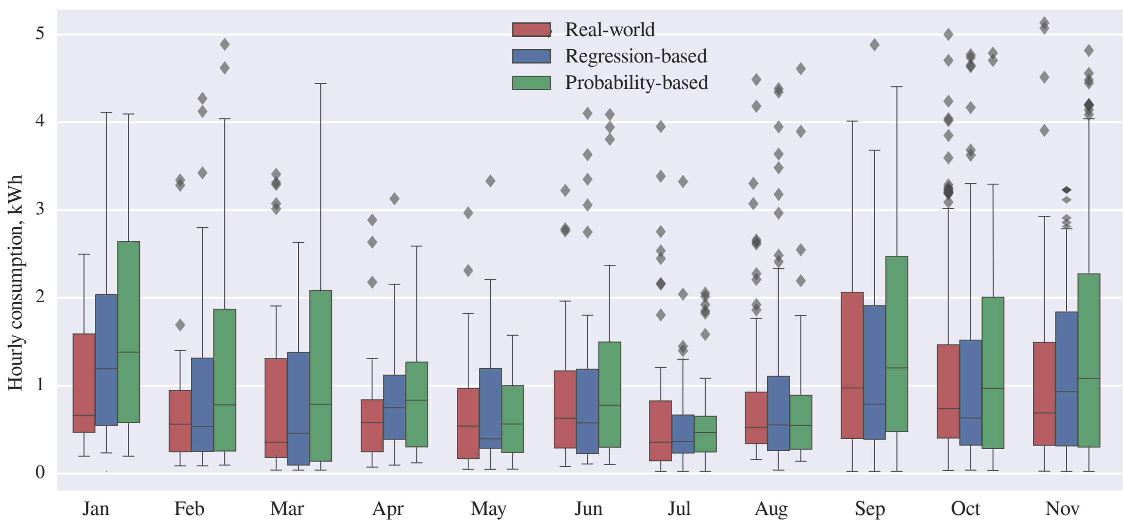

Figure 14: Comparison of summary statistic

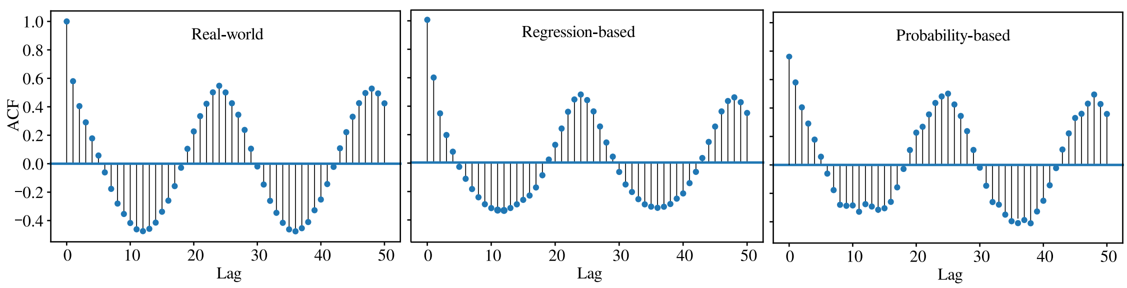

Figure 15: Comparison of normalized time-series auto-correlation

more computer time than the normal K-means clustering (see Figure 14). The overall time used by the probability-based method is higher than the regression-based method as it consists of five mandatory steps in the whole training process.

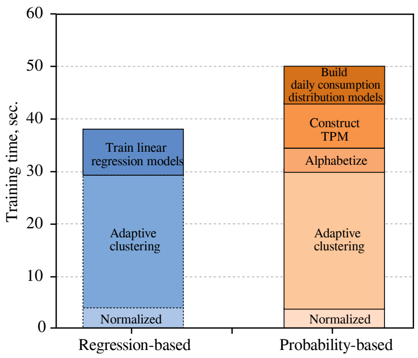

Figure 16: Comparison of training times

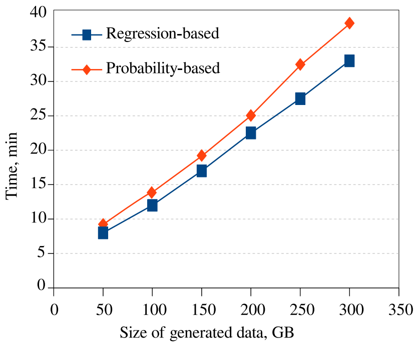

Figure 17: Size-up of generating data sets

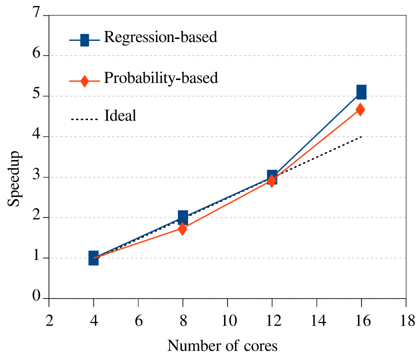

Figure 18: Speedup of generating 100GB data

In the following, we will evaluate the scalability of generating time series on Spark. It should be remembered that the models created by the training process are distributed to the workers by broadcasting in Spark. Map-only tasks are conducted to generate time series data in parallel. We performed the following two experiments to evaluate the scalability, *size-up* and *speedup*.

In the size-up experiment, we used a total of four nodes (16 cores) to generate the data, but scaled the generated data set from 50 to 300 GB. Figure 17 shows the execution times that resulted. The results demonstrate that the time scaled well with the amount of data, almost linearly.

In the speedup experiment, we scaled the cores from 4 to 16 to generate a fixed-size data set ($100GB$), and measure the execution times. The speedup is defined by the following equation:

$$speedup = \frac{T_4}{T_n} \qquad (15)$$

where T_n is the execution time for$n$ cores ($n = 4, 8, 12$ and 16). Figure 18 shows the results. According to the result, both of the proposed methods can achieve good speedup, and the speedup is super linear when the cores increase to 16. In both experiments, the two methods are quite efficient in terms of running time and scalability as they run map-only jobs. The performance of the regression-based method is slightly better than that of the probability-based method, mainly due to the cost of constructing the alphabet sequence by random walk. As the size of the TPM is very large, $n^2 * m = 20^2 * 535 = 214,000$ ($n = 20, m = 365 - 1$) where n is the number of states (representative patterns) and m is the number of days in one year, the cost of lookups on TPM is substantial.

5.4. Discussion

In summary, the proposed data generators were able to generate realistic time series data with good performance, and the generated data have characteristics that are comparable to the real-world data in terms of patterns and statistical information. The two methods were supervised machine learning methods that require real-world data sets as the seed for generating realistic data sets. Our study indicates that clustering is a good way to preserve consumption patterns and customer segmentation information. The two methods differ in the following way: one uses prediction for consumption data simulation, while the other uses statistics and probability. For the regression-based method, the accuracy of the simulation is highly dependent on the prediction model. In reality, it is often difficult to establish an accurate predictive model as it is affected by many variables, such as building type, household characteristics, weather conditions. In this paper, we simply used the autoregressive model for the prediction where only the time series data are considered. There are other prediction models that can be used for this simulation if the relevant socioeconomic data are available, for example, the *periodic auto-regression with exogenous variables model (PARX)* [32] which takes into account other factors that may affect energy consumption. In contrast, the probability-based method simulates the real-world consumption based on the statistical parameters of the data. In our experiment, this approach was still able to generate satisfactory results. As it is often difficult to obtain socioeconomic data, largely due to data privacy issues, the probability-based method can be a good alternative for simulating real-world consumption data.

The implementation of these methods includes the training and generation programming. The training process in both methods requires multiple steps. Comparatively, the regression-based method would require less human and computer effort if the optional steps for optimisation, normalisation, and clustering were omitted. The most time-consuming step is clustering for the training, which is for determining household groups or representative pattern groups. Depending on the size of the seeded data, the training programming may not have to be implemented using a distributed computing programming framework, such as Spark, as in this work. The training process is run only once, but the resulting models will be re-used many times. The implementation of the data generation program is relatively easier, as it is a map-only program on Spark. The parameters or models for data generation are also distributed to mappers during the runtime through broadcasting and are kept in memory for generating data for better efficiency. Using a distributed computing framework makes it possible to generate data in parallel. There are other alternatives for parallel data generation such as multi-threading. However, a cluster-based program would be the best option for generating large-scale data sets of the order of tera/petabytes. Big data sets are often needed for benchmarking big data management systems, e.g., [31].

6. Conclusions and Future Work

Scalable realistic consumption time series data are often needed for system benchmarking in software engineering and for building performance evaluation in civil engineering. In this paper, we have presented two very different data generators that can accurately simulate real-world fine-grain energy consumption time series. The proposed methods are both supervised machine learning methods that include a training and a data generation process. However, they are based on different techniques: one is regression-based and the other is probability-based. We have detailed how to create data models, and how to use the models to generate synthetic data sets. We have proposed optimisation techniques for better simulating the real-world data, such as preserving cluster segmentation information, and have implemented the data generator on Spark so as to generate data in parallel. We have evaluated the proposed methods comprehensively and compared the two methods. The results have shown that the proposed methods have the ability to simulate realistic energy consumption data, and the implemented data generators have good performance for generating large-scale data sets.

In future work, we will add more features to improve data generation models. For example, the regression-based method can use weather conditions (e.g., outdoor temperatures), and wider seasonality (e.g., the seasons of a year). In addition, we will refine the data generators to make them easy to use for generating different consumption data, such as water, gas, or heat.

Acknowledgements. This research is part of the ClairCity project (http://www.claircity.eu) funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 689289, and by UCN-FOU funding (Project-6/2016-17).

References

[1] V. Abeykoon, N. Kankanamdurage, A. Senevirathna, P. Ranaweera, and R. Udawalpola, R. Electrical Devices Identification through Power Consumption using Machine Learning Techniques. International Journal of Simulation–Systems, Science & Technology, 17(26), 2016.

[2] M.S. Ahmed, A. Mohamed, R.Z. Homod, and H. Shareef, A.H. Sabry, and K.B. Khalid. Smart plug prototype for monitoring electrical appliances in Home Energy Management System. In Proc. of IEEE Student Conference on Research and Development (SCOReD), pp. 32–36, 2015.

[3] D.J. Aigner, C. Sorooshian, and P. Kerwin. Conditional demand analysis for estimating residential end-use load profiles. The Energy Journal, 5(3):81–97, 1984.

[4] P.L. Anderson, M.M. Meerschaert, K. Zhang, Forecasting with Prediction Intervals for Periodic Autoregressive Moving Average Models. Journal of Time Series Analysis, 34(2)(2013)187–193.

[5] M. Arlitt, M. Marwah, G. Bellala, A. Shah, J. Healey, B. Vandiver, IoTABench: An Internet of Things Analytics Benchmark, in: Proc. of the 6th ACM/SPEC International Conference on Performance Engineering, 2015, pp. 133–144.

[6] O. Ardakanian, N. Koochakzadeh, R. P. Singh, L. Golab, and S.Keshav,Computing Electricity Consumption Profiles from Household Smart Me-ter Data. In Proc. of the EDBT Workshop on Energy Data Management(EnDM), pp. 140–147, 2014.

[7] N. Batra, O. Parson, M. Berges, A. Singh, and A. Rogers. A comparison of non-intrusive load monitoring methods for commercial and residential buildings. arXiv preprint arXiv:1408.6595, 2014.

[8] Y. Bengio, Learning deep architectures for AI, Found. Trends Mach. Learn. 2

[9] B.J. Birt, G. R. Newsham, I. Beausoleil-Morrison, M. M. Armstrong, N. Saldanha, and I. H. Rowlands. Disaggregating categories of electrical energy end-use from whole-house hourly data. Energy and Buildings, 50:93–102, 2012.

[10] K. Black, Business Statistics: For Contemporary Decision Making, John Wiley & Sons 2011.

[11] J.G. De Gooijer, R.J. Hyndman, 25 Years of Time Series Forecasting, International Journal of Forecasting, 22(3)(2006) 443–473.

[12] S. Dolnicar. A Review of Unquestioned Standards in Using Cluster Analysis for Data-driven Market Segmentation. Proc. of the Australian and New Zealand Marketing Academy Conference, 2002.

[13] B. Dong, C. Cao, and S.E. Lee. Applying support vector machines to predict building energy consumption in tropical region. Energy and Buildings, 37(5):545-553, 2005.

[14] S. Fan, R. Hyndman. Short-term load forecasting based on a semi-parametric

[15] S. Bissey, S. Jacques, and J.C. Le Bunetel. The Fuzzy Logic Method to Efficiently Optimize Electricity Consumption in Individual Housing. Energies 10(11): 1701, 2017.

[16] L. Chuan, and A. Ukil. Modeling and validation of electrical load profiling in residential buildings in Singapore. IEEE Transactions on Power Systems, 30(5):2800–2809, 2015.

[17] C. Fan, F. Xiao, and Y. Zhao. A short-term building cooling load prediction method using deep learning algorithms. Applied energy, 195:222–233, 2017.

[18] General Data Protection Regulation (GDPR). Avail. at https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en as of 2018.06.02.

[19] P. Gianniou, X. Liu, A. Heller, P. S. Nielsen, and C. Rode . Clustering-based Analysis for Residential District Heating Data. Journal of Energy Conversion & Management, vol. 165, pp. 840–850, 2018.

[20] P. A. Jaskowiak, R. J. Campello, and I. G. Costa. Proximity measures for clustering gene expression microarray data: a validation methodology and a comparative analysis. IEEE/ACM transactions on computational biology and bioinformatics, 10(4):845–857, 2013.

[21] G.M. Jenkins. AutoregressiveMoving Average (ARMA) Models. Encyclopedia of Statistical Sciences, 1982.

[22] N. Iftikhar, X. Liu, S. Danalachi, F.E. Nordbjerg, J.H. Vollesen, A Scalable Smart Meter Data Generator Using Spark, in: Proc. of OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", 2017, pp. 21–36.

[23] ISSDA, `www.ucd.ie/issda/data/commissionforenergyregulationcer` as of 2018-04-20.

[24] J. Kwac, J. Flora, R. Rajagopal, Household energy consumption segmentation using hourly data, IEEE Transactions on Smart Grid, 5(1)(2014) 420–430.

[25] K.D. Lawrence, R.K. Klimberg, S.M. Lawrence, Fundamentals of Forecasting using Excel, Industrial Press Inc. (2009).

[26] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. Nature, 321:436–44, 2015.

[27] Q. Li, P. Ren, and Q. Meng. Prediction model of annual energy consumption of residential buildings. Proc. of the Advances in Energy Engineering (ICAEE) Conference, pp. 223–226, 2010.

[28] T.W. Liao, Clustering of Time Series Data–A Survey. Pattern Recognition, 38(11)(2005) 1857–1874.

[29] X. Liu, and M. C. Papaefthymiou. A markov chain sequence generator for power macromodeling. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 23(7):1048-1062, 2004.

[30] X. Liu, L. Golab, W. Golab, I.F. Ilyas, Benchmarking Smart Meter Data Analytics, in: Proc. of the 18th International Conference on Extending Database Technology, 2015, pp. 385–396.

[31] X. Liu, L. Golab, W. Golab, I.F. Ilyas, S. Jin, Smart Meter Data Analytics: Systems, Algorithms, and Benchmarking, ACM Transactions on Database Systems (TODS), 42(1), 2017.

[32] X. Liu, P. S. Nielsen. An ICT-Solution for Smart Meter Data Analytics. Journal of Energy, 115(3):1710–1722, 2016.

[33] S. Makonin, B. Ellert, I.V. Bajic, and F. Popowich. Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014. Scientific data, 3, 160037, 2016.

[34] A. Marszal-Pomianowska, P. Heiselberg, and O.K. Larsen. Household electricity demand profiles – A high-resolution load model to facilitate modelling of energy flexible buildings. Energy, 103:487-501, 2016.

[35] E. Mocanu, P.H. Nguyen, M. Gibescu, and W.L. Kling, W. L. Deep learning for estimating building energy consumption. Sustainable Energy, Grids and Networks, 6, 91–99, 2016..

[36] B. O. Ngoko, H. Sugihara, and T. Funaki. Synthetic generation of high temporal resolution solar radiation data using Markov models. Solar Energy, 103: 160-170, 2014.

[37] A. Okcan, M. Riedewald, Processing theta-joins using MapReduce, in: Proc. of SIGMOD, 2011, pp. 949–960.

[38] M. Parsian, Data Algorithms: Recipes for Scaling Up with Hadoop and Spark, O'Reilly Media, Inc. 2015.

[39] N. Pflugradt, U. Muntwyler, Synthesizing residential load profiles using behavior simulation, Energy Procedia, 122(2017) 655–660.

[40] M. Pipattanasomporn, M. Kuzlu, S. Rahman, and Y. Teklu, Y. Load Profiles of Selected Major Household Appliances and Their Demand Response Opportunities. IEEE Transaction Smart Grid 5:742–750, 2014.

[41] I. Richardson, M. Thomson, D. Infield, and C. Clifford. Domestic electricity use: A high-resolution energy demand model. Energy and buildings, 42(10):1878–1887, 2010.

[42] Smart Meter From Wikipedia, en.wikipedia.org/wiki/Smart_meter as of 2018-04-20.
[43] P. Street. Dataport: the world's largest energy data resource. Pecan Street Inc., 2015.
[44] N. Tewathia. Determinants of the household electricity consumption: A case study of Delhi. International Journal of Energy Economics and Policy, 4(3):337–348.
[45] R. Weiers, Introduction to Business Statistics. Cengage Learning, 2010.
[46] J. Wu, Advances in K-means Clustering: A Data Mining Thinking. Springer Science & Business Media, 2012.
[47] G.P. Zhang, Time Series Forecasting using a Hybrid ARIMA and Neural Network Model. Neurocomputing, 50(2003) 159–175.
[48] G.P. Zhang, M. Qi, Neural Network Forecasting for Seasonal and Trend Time Series, European Journal of Operational Research, 160(2)(2005) 501–514.
[49] J. Zico Kolter, and Matthew J. Johnson. REDD: A public data set for energy disaggregation research. In proceedings of the SustKDD workshop on Data Mining Applications in Sustainability, 2011.

Dr.Xiufeng Liu, Technical University of Denmark (Xiuli@dtu.dk)
Xiufeng Liu received his PhD in Computer Science from Aalborg University, Denmark in 2012. Between 2013 and 2014, he worked as a Postdoctoral researcher at Waterloo University, Canada, and since January 2015 he has been working as a Postdoctoral researcher in Technical University of Denmark. His research areas are smart meter data analytics, data warehousing, and big data.

Dr. Nadeem Iftikhar, University College of Northern Denmark (naif@ucn.dk)
Nadeem Iftikhar is an Associate Professor at University College of Northern Denmark. He received the BS degree in Applied Mathematics and Computer Science from Eastern Mediterranean University, Famagusta, Turkish Republic of Northern Cyprus, in 1994, followed by the MS degree in Information Science from University of New South Wales, Sydney, Australia, in 1997 and the Ph.D. degree in Computer Science from Aalborg University, Aalborg, Denmark, in 2011. His research activity and published work concerns the area of data warehousing, with a particular emphasis on OLAP and ETL, as well as big data analytics. He is an IEEE member.

Dr. Huan Huo, University of Technology Sydney, Australia (huan.huo@uts.edu.au**)**
Dr. Huan HUO receives her Ph.D. in Computer Software and Theory from Northeastern University and currently works as an associate professor in Optical-Electrical and Computer Engineering School at the University of Shanghai for Science and Technology. Her research interests include cloud data management technology, data stream query optimization, XML data management technology, etc.

Dr. Rongling Li, Technical University of Denmark (liron@byg.dtu.dk**)**

Rongling is a postdoctoral researcher at Civil engineering department at Technical University of Denmark
.

Dr. Per Sieverts Nielsen, Technical University of Denmark (pernn@dtu.dk**)**
Per has been working in the energy modelling and sustainability area since 1990. He is senior researcher in the System Analysis Division at the Department of Management Engineering at the Technical University of Denmark. He has participated in a range of international research programmes and multi-disciplinary research programmes. Currently Per is working on various projects in energy systems integration in particular "smart cities" development. He is currently supervisor of three PhD students and Co-supervisor of another two PhD students.

Dr.Xiufeng Liu, Technical University of Denmark (Xiuli@dtu.dk)

Dr. Nadeem Iftikhar, University College of Northern Denmark (naif@ucn.dk)

Dr. Huan Huo, University of Technology Sydney, Australia (huan.huo@uts.edu.au)

Dr. Rongling Li, Technical University of Denmark (liron@byg.dtu.dk**)**

Dr. Per Sieverts Nielsen, Technical University of Denmark (pernn@dtu.dk**)**