Effects of Data Cleansing on Load Prediction Algorithms

Boye A. Høverstad, Axel Tidemann and Helge Langseth

Department of Computer and Information Science, Norwegian University of Science and Technology,
Trondheim, Norway
Email: { hoversta, tidemann, helgel }@idi.ntnu.no

***Abstract*—The rollout of advanced metering infrastructure that is planned in many countries worldwide will lead to a massive inflow of data from moderately reliable sensory equipment. In principle, this will make intelligent and automated planning and operation possible at an increasingly finer scale in the electric grid. However, errors can creep into the meter data, either from faulty sensors or during transmission from the meters to the database. This work studies the role of data cleansing as a preprocessing step for short-term (24-hour) power load prediction. We focus on cleansing and prediction at several levels of granularity, from the transmission level via distribution substations down to single households.**

We believe that preprocessing filters such as cleansing should lead to more robustness and/or precision in the subsequent processing step. However, load cleansing frameworks tend to make the popular assumption of normally and independently distributed noise in the time series. We show that this is incorrect at the diurnal level, due to the characteristic pattern of power consumption, with two peak loads during daytime and a nighttime trough. Moreover, we present empirical evidence that a preprocessing step based on this assumption fails to contribute positively to the performance of the subsequent prediction step. To rectify this problem, we suggest to subtract the average power load consumption in a given period before cleansing. We present empirical evidence that this improves the robustness and efficiency of load cleansing as a preprocessing step. Data cleansing and load prediction is performed by a system that searches out parameters using an evolutionary approach.

I. Introduction

A. Background

Power load forecasting has been of interest for quite some time and a plethora of methods have been attempted to obtain accurate forecasting results. According to the time scale they operate on, load forecasting systems are often categorized as long-term, medium-term or short-term systems. Long term (up to 50 years ahead) forecasting is required for capacity planning and maintenance scheduling while medium term (up to 10 years ahead) forecasting are needed for power system operation and planning. The importance of short term (up to a week ahead) forecasting increased after deregulation of the power market, allowing competition between the buying and selling parties. Short-term load forecasts are needed by market operators for determining the day-ahead market prices and by the market participants for preparing bids, and gains even more importance as the grid is transformed to a smart grid. As the grid becomes more dynamic with the introduction of various local power generators and storage units, predictions are essential to make cost-effective, real time decisions while avoiding voltage band violations. Various prediction strategies are presented in the literature; Bunn and Farmer [1] review the earlier load forecasting models while more recent approaches are summarized by Alfares et al. [2].

The systematic (also denoted "predictable") part of the load data signal consists of patterns repeating in a daily, weekly and seasonal rythm. Further, this signal is obscured by various activities and conditions pertinent to the grid, which is commonly referred to as "noise"; hence, load data may be quite complex and at least partly affected by external variables (e.g. such as weather profiles, history of prices and unusual/abnormal consumption behaviors, for example due to disasters, strikes, and TV events). The quality of the data is another factor accentuating the complexity of the forecasting process. An important problem in load data analysis is therefore *data validation*, where the task is to distinguish between data corruption and a change in data pattern due to a random event or periodical patterns. The overall process of data validation, noise removal and preparation for further analysis is often referred to as *data cleansing*.

This paper focuses on a short-term (24 hours ahead) forecasting method that integrates data cleansing and load prediction. The majority of approaches in the literature examine either of these in isolation, but fail to properly consider how the two are intricately intertwined. We examine four prediction models (Autoregression, Indexed Autoregression, Echo State Networks, Wavelets) and three levels of granularity of load prediction (single-user, distribution substation, transmission) to study how the algorithms and datasets influence the effect of data cleansing. In order to make the system as automatic as possible, a genetic algorithm is run in order to parameterize both the data cleansing and the prediction algorithms.

B. Related Work

From a methodological point of view, load prediction approaches have been broadly categorized into *statistical* (parametric) and *artificial intelligence-based* (nonparametric) techniques [3]. A wide variety of statistical methods have been employed for short term load forecasting including regression and time series techniques. Time series models include autoregression (AR) models, autoregression moving-average (ARMA) models and autoregression integrated moving-average (ARIMA) models. ARIMA is perhaps the most used method in this group (see [4] for an example),

but it uses only past load data and does not explicitly use exogenous data. Autoregression moving-average with exogenous data (ARMAX, e.g., [5]) may also be included into this group.

Artificial intelligence techniques include, among others, fuzzy logic, genetic algorithms and neural networks. These are well-studied non-linear techniques that are often combined. For example, [6] uses ANNs and a fuzzy expert system in a two-staged forecasting process. ANNs were used in the first stage to train the forecasting algorithm using past load patterns. The forecasted load was then modified to deal with possible load variation due to exogenous factors such as temperature and the load behavior of holidays. In a similar spirit, Srinivasan et al. [7] developed a method integrating fuzzy logic, neural networks and expert systems.

Load data consists of a time series of load values partially influenced by a number of factors such as day-of-week, season, economic factors, and weather. Additionally, load data is altered through random events such as outages, device failures, and network communication errors, which motivates a principal approach to load data cleansing. In the scarce literature on this topic, we are inspired by Chen et al. [9]. They model the data generation process underlying the load curve data as a continuous function. Two nonparametric regression methods, B-spline and Kernel smoothing, have been used to estimate this function. Based on this estimate a point-wise interval confidence is established. Data points outside this interval are considered corrupted. The approach introduces user involvement into the decision of the smoothness parameter, which is undesirable for two reasons: 1) we argue that the data cleansing process does more than simply removing outliers – it transforms the data in ways optimal for the data analysis task at hand. What this explicitly means in a particular situation may be unknown, and therefore not obtainable via user interaction. 2) A research goal is that the system can configure itself, since load prediction will be performed on different data sets with different inherent characteristics.

II. Materials and Methods

This paper presents an approach that employs data cleansing and model training to do load prediction. A key issue when employing any method is knowing what parameters will yield the best performance. Setting the parameters often requires experience and intuition. Similar to [8], we use an *evolutionary approach* to set these parameters. This approach also takes into account dependability of parameter values across data cleansing and model training. In other words, the genetic algorithm is used to parameterize data cleansing and model training.

A genetic algorithm (GA) works by trying out many different solutions to a problem. In the initial population, all the different solutions are generated randomly. The solutions that have the lowest prediction error (which in the GA terminology are known as the "individuals" that are most "fit") are passed on to the next generation, where new "children" are created from the best individuals by genetic crossover (combining parents' genes) and mutation (randomly altering a gene). In problems where the parameter space is too large to be searched exhaustively, and the structure of the problem is too complex to be solved in polynomial time, genetic algorithms have been shown to find good approximative solutions [10].

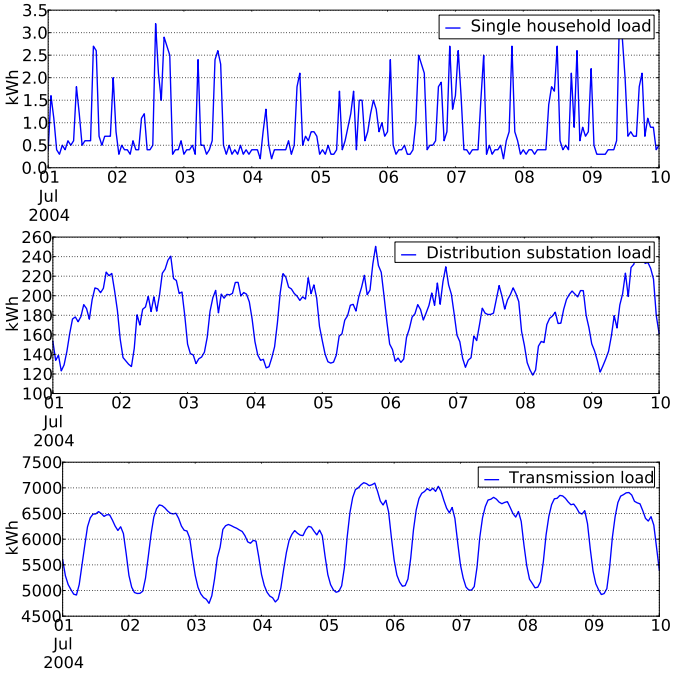


Fig. 1: 10 days of power load data at various levels of resolution. From top to bottom: single household, distribution substation, and transmission level.

In the experiments reported in this paper, the genome of an individual codes the parameters of the load cleansing and prediction model and the fitness of an individual is the inverse of its prediction error. The remainder of this section will describe the experimental setup we have used to examine the effect of data cleansing in relation to load prediction. First, the dataset will be presented, before more details behind data cleansing and model training are given. The genome is then specified, tying everything together in the genetic algorithm.

A. Dataset

Our data comes from two sources: a dataset from an earlier research project from SINTEF Energy Research in cooperation with several Norwegian utility companies [11]. The original dataset contains load measurements from a small number of Norwegian cities and municipalities, in the period 2004-2006. The dataset includes an hourly temperature reading for the feeder. We thus expect the measured temperatures to be a fairly accurate representation of the conditions experienced at each household, even though some local variations in temperature are not accounted for. The inputs to our system are hourly readings of *load* and *temperature*. In 2006, the average Norwegian household spent approximately 40-45% of its total energy consumption on space heating, and another 10-15% on water heating [12]. Temperature is thus an environmental factor that is expected to have a large impact on power consumption.

Each load time series represents hourly power load consumption in kWh for a single anonymized household. There were approximately 2400 meters connected to the feeder with temperature readings. We restricted our focus to a small

number of households that were pseudo-randomly selected based on the following requirements.

Part of the dataset contained readings from meters with 1 kWh resolution. These were discarded, such that only measurements from meters with at least 0.01 kWh resolution were used. Next, we selected the meters for which we had a minimum of missing data over the three-year measurement period. This resulted in a selection of 150 meters (equal to a large urban distribution substation) with consecutive readings over a training period March 2004 - July 2005, and a somewhat shorter test period October 2005 - October 2006.

The other dataset is transmission level load data from British Columbia (BC Hydro), from January 2004 throughout December 2010[1].

From these two datasets, we work on three different levels of granularity of the data, as shown in Fig. 1. These are:

1) Single user: Data from a randomly selected household, obtained from an Advanced Metering System (AMS).

2) Distribution substation: Aggregated data from 150 households, i.e. a substation in a densely populated urban area. The resulting time series shows daily and weekly trends, normally with two discernible daily peaks, yet it is much more noisy and less periodic than a typical power load time series at e.g. the transmission level.

3) Transmission: Canadian transmission level data. For this dataset, temperature is not available. The transmission system consists of over 18,000 km of lines and underwater submarine cables and 292 substations.

B. Cleansing Model

We used the B-spline approach of Chen et al. [9] to clean both temperature and load data before feeding these to the load prediction model. This method approximates the observed data points $\boldsymbol{y}$ with a smooth curve as described by the spline:

$$m(t) = \boldsymbol{c}^T \boldsymbol{\phi}(t) \tag{1}$$

where $\boldsymbol{\phi}(t)$ is a vector of B-spline basis functions, and the coefficient vector $\boldsymbol{c}$ is estimated from the observations $\boldsymbol{y}$ at time t. Similar to [9], we use a standard smoothing spline with a full set of knots and a regularization term. The spline is found by minimizing this *penalized sum of squares*:

$$\mathrm{PENSSE}(\lambda) = \sum_t (y_t - m(t))^2 + \lambda \int (m''(t))^2 \,\mathrm{d}t \tag{2}$$

The first term rewards closeness to the data, while the regularization term penalizes the curvature of the spline. The trade-off between goodness of fit and spline smoothness is controlled by the *smoothing parameter* λ. A more thorough treatise of the fitting procedure can be found in Hastie et al. [13] or Ramsay et al. [14].

Given λ and the corresponding spline, a confidence interval around can be determined based on an estimate of the variance of the data to the fit. Under assumptions of normality and independence, the estimated predicted error reduces to [9]:

$$s^2\{\mathrm{pred}\} = \sigma^2 + \mathrm{diag}(\sigma^2 \boldsymbol{S}\boldsymbol{S}^T) \tag{3}$$

where σ^2 is approximated by the mean square error of the data to the fit, and $\boldsymbol{S}$ is the smoothing matrix found by solving 2. Given a *Z-score* parameter z, points outside of the interval $m(t) \pm z \cdot s_t^2\{\mathrm{pred}\}$ are identified as outliers.

For the work presented here, it is important to note that the cleansing algorithm is controlled by two key parameters: the smoothness of the B-spline curve (λ), and the width of the confidence interval (z).

The identified outliers must be replaced by a value that is better by some measure, so that the cleansed data can be used for prediction. In the work described here, *better* means an improvement in the subsequent prediction, irrespective of the underlying cause of the outlying observation. In the experiments described in the next section, we replace outlier values with the corresponding upper or lower bound of the confidence interval.

Note that these two parameters are highly interdependent. A wide confidence interval will weaken the impact of the smoothing parameter on the cleansing process, and vice versa. This is especially true when outliers are not only identified but also replaced with a cleansed estimate. In our experiments, both the smoothness and the Z-score are part of the genome, so the genetic algorithm may adjust both the fraction of observed data that are replaced with approximations, and the proximity of the replacement values to the observed curve.

Both the approach adopted here, and other methods [15], perform a nonparametric regression to the power load time series, upon which outliers are identified by making the typical assumption of independent and identically distributed Gaussian noise. This is reasonable when domain knowledge is scarce or there are complex patterns in the data. However, both power load and temperature time series often have typical diurnal patterns: there are normally peaks in the morning and afternoon, and a trough during nighttime. As a consequence, the gradient leading up to the morning peak hour and following afternoon peak hour are typically much sharper than for the rest of the day. A non-parametric regression will thus tend to have high error during these periods.

The result of performing cleansing with such a method is typically that peak hour loads are identified as outliers. As a general rule, this is very unfortunate, since precise identification (and prediction) of peak load is of utmost importance for safe and efficient grid operation.

In order to rectify this problem, at least two approaches can be envisaged: domain knowledge can be included in the regression, or the data can be transformed prior to performing the regression. In this paper we follow the latter path. The typical daily pattern is removed from the time series in two steps, as illustrated in Fig. 2: first, the historical average load of each calendar day is subtracted from every data point for that day. This produces a signal with zero mean for every calendar day in the training data. The effect is to eliminate seasonal variations, yielding a signal with little difference between summer and winter days.

[1] http://transmission.bchydro.com/transmission_system/balancing_authority_load_data/historical_transmission_data.htm

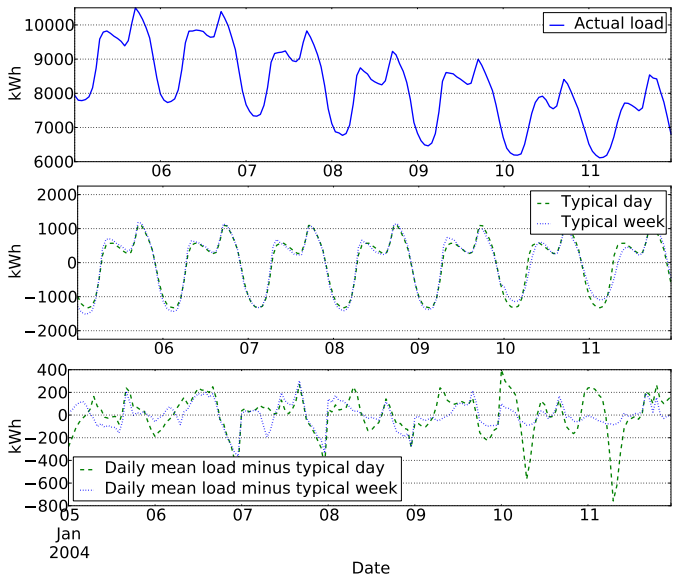


Fig. 2: Transformation of the load time series prior to outlier detection and removal. The training data (top figure) is shifted by subtracting the daily mean, after which a "typical day" or "typical week" is calculated (middle figure). Subtracting the typical day/week from the shifted data produces the signal used for regression (bottom figure).

Next, a "prototypical" day is estimated using one of two methods, termed *daily* and *weekly* in the remainder of this paper. The daily prototype method estimates a typical 24-hour period as the mean of each clock hour for all days in the training data. The weekly prototype further refines this by calculating a 24-hour prototype for each weekday, producing a 168-hour series representing a "typical week". As expected, the difference between these two approaches is larger for Saturdays and Sundays than for weekdays. The transformed series from the first step is then subtracted its respective typical day, to produce the signal on which regression and outlier detection is performed. Finally, the cleansed signal is retrieved by reversing the above procedure, before being fed to the prediction algorithm.

C. Prediction Models

In our framework, we use four different models to predict 24 hours into the future. These will now be presented. In accordance with Table I, the parameters of each model is specified. The genetic algorithm searches for the parameter values for each method. Common to all the models is the *hindsight* parameter, which defines how much history the model uses when it predicts 24 hours into the future.

1) Autoregressive model: An autoregressive model (AR) is a linear predictor. Its parameter is the *order* of the AR, i.e. how many previous values contribute to the output. The calculation of how much these previous values contribute to the output is typically calculated by the least squares method. In this work, we have chosen to report the results of using a standard AR model rather than an ARIMA or ARIMAX for two reasons. First, we expect the load cleansing process to reduce some of the noise in the signal fed to the prediction model, thus eliminating the need for the MA step. Second, the simplicity of the AR model allows for the use of considerably higher orders, which we believe is favorable in the domain of power load prediction. This is confirmed by load time series autocorrelation plots (not shown).

2) Indexed autoregressive model: In order to further highlight the effect of the load cleansing process, we developed a simple extension to the AR model that exploits domain knowledge of the data: the load in the previous hour is obviously important to determine current load, but from inspecting the data it becomes evident that load data is highly recurrent at each day, and also at each week.

The indexed autoregressive model is restricted to using only observations from these periods. In other words, depending on the genome, the model performs an autoregression using only the observations $\{1\}$, $\{1, 24\}$, or $\{1, 24, 168\}$ hours prior to each timestep.

3) Echo State Network: An Echo State Network (ESN) [16] is a type of *reservoir model*, a class of recurrent neural network models where the main computational unit is a large collection of stochastically interconnected nodes. This is known as the *reservoir*. The prototypical ESN consists of two or three units: an optional set of input nodes, the reservoir containing nonlinear nodes and fixed interconnections, and a number of linear readout nodes. The input nodes are fully connected to the reservoir, and the reservoir is in turn fully connected to the output nodes. Finally, for signal generation tasks, the output from the readout nodes are fed back to the reservoir. Learning is typically done as a linear regression on the incoming connections to the output layer, i.e. the training time is considerably faster compared to traditional networks trained by backpropagation.

Reservoir models have been shown to perform very well on pattern recognition and classification [17], and time series prediction tasks [16]. ESNs have also previously been employed for short-term power load prediction [18], [19], [20].

The overall dynamics of an ESN is governed by several key parameters (see table I). Importantly, these parameters are both interdependent and dependent on the problem at hand. For instance, the leakage rate and spectral radius will affect both each other and the memory length of the network [21].

We train the ESN to predict 24 hours into the future, following a spin-up and training phase where it is driven by historical loads and temperatures. Power load time series typically have a strong diurnal periodicity. The load consumption 24 hours ago is an important indicator at the expected load now. Our network thus receives three inputs during training and spin-up: the current temperature, the observed load 24 hours ago, and the observed load in the previous hour. When switching to prediction mode, the input representing the observed load in the previous hour is replaced with a feedback of the ESN prediction. In this regard, the network works as a signal generation model with external inputs. The temperatures can be interpreted as an optimistic approximation of the weather forecast. In previous work, this has been shown to be an acceptable simplification [22].

TABLE I: List of the genes that specify the different parameters of the system.

Model	Gene	Range
Data cleansing	Load smoothing	0.001 - 1000 (log)
Data cleansing	Load Z-score	0.1 - 3
Data cleansing	Temperature smoothing	0.001 - 1000 (log)
Data cleansing	Temperature Z-score	0.1 - 3
ESN	Network size	10 - 500
ESN	Leak rate	0 - 2
ESN	Input scaling	0.1 - 0.75
ESN	Bias scaling	0 - 1
ESN	Spectral radius	0.5 - 2
AR	Order	100-200
AR Index	Indices	1, 24, 168
Wavelet	Scale	1-10
Dataset	Hindsight (hours)	24, 72, 168, 336, 672
System	Seed	5 random numbers

4) Wavelets: The wavelet transform is a representation of a signal in different scales that is obtained by convolving (i.e. scaling and shifting) the input signal with a wavelet (i.e. "small wave"). There is a variety of wavelets used (we employ the Haar wavelet), the requirement is that they integrate to zero. The wavelet transform is ideal for compression, since it retains just the exact amount of information to reconstruct the original signal. However, by using a redundant wavelet transform (also known as the "à trous transform") one obtains continuous signals at each scale that can be used for prediction. The redundant wavelet transformation thus yields several band-pass filtered components and a low-pass filtered signal. A subset of the values in each of these scales is used to predict the next value in the input signal. This can be done by any linear or non-linear predictor, we employ linear regression, using least squares. In our case, the target signal is 24 hours into the future. This prediction approach is described in detail in [23], and has been applied for load forecasting [24]. However, in the latter work, they predicted 1 hour into the future. Our focus is on 24 hour prediction, which is obviously more difficult to do. Furthermore, their system is based on individual fine-tuning of the parameters performed by an operator, whereas our system solves this automatically by evolution (see next section).

The only parameter for the wavelet is the number of *scales* the signal should be decomposed into. In some ways, this approach is similar to ESNs, since it transforms a 1-D input signal into a higher dimensional space, where it becomes easier to make the data points linearly separable. What separates the wavelet-based prediction from the ESN algorithm is that the wavelet transformation is not a stochastic process. The wavelet transform is preferred over the Fourier transform, since the former conserves both time and frequency information, whereas the Fourier transform only preserves frequency information.

D. Evolution Model

The genome contains the parameters of the data cleansing, model training, and the dataset, see Table I. The genes were mapped to their respective allele values upon fitness evaluation. The smoothing and Z-score parameters used by the data cleansing algorithm are set separately for power load and temperature. The smoothing genes were represented in logarithmic space in order to allow for higher resolution in the lower part of the range. Data length specifies how many data points (i.e. hours) are part of the training set for the genome. The model is trained to predict 24 hours into the future, and the gene defines how much of the history (i.e. length of the training signal) is used to train the model. A system variable is part of the genome, namely the *seed* of the random number generator. The seed is part of the genome because the ESN is not specified in a deterministic manner, due to its inherent randomly generated network architecture. In order to preserve the best individual from each generation, the seed must be set prior to the network generation, or else an inferior phenotype could be created from the same genome. All genes were represented in the GA as floating point numbers in the range $[0, 100]$.

The reproduction operators were single-point crossover and Gaussian mutation. Finally, the evolution used tournament selection with sigma truncation scaling. Prediction error was calculated using the mean square error (RMSE). We used RMSE rather than MAPE, since MAPE is unstable for values close to zero, which frequently occur at the single user level. The genetic algorithm was run on the training set as follows. Seven dates were randomly selected from the training set at the beginning of the evolutionary run. Each individual was trained to predict the following 24 hours of that date. Fitness was then calculated as $1/(1 + \text{RMSE})$. The network was fed with load and temperature data cleansed using the genetically specified parameters. However, prediction error and fitness were calculated by comparing to the observed load *without* cleansing. After each generation, three of the seven dates were swapped out with new randomly selected dates. This was done to make the models able to generalize.

III. Experiments

Experiments were performed to test our hypothesis that a transformation of the data prior to outlier detection (i.e. to eliminate or reduce typical patterns at the prediction time scale), will make the assumption of independence and normality in the noise more appropriate. This will result in a data cleansing that improves the overall performance and robustness of the prediction system as a whole.

For each of the 3 datasets we evaluated the four prediction algorithms, both with and without data cleansing to measure its efficacy. The evolutionary search was run 30 times on each combination of data cleansing method, prediction algorithm and dataset, with different initial random seeds, in order to see whether the system would consistently converge on a set of optimal cleansing parameter values. We evolved populations of 100 individuals for 100 generations, using mutation rate 0.2 and crossover rate 0.5.

After evolution, the best genomes from each run were examined on a validation dataset, where the prediction error over a larger number of predictions was recorded (approximately 6 months, depending on the dataset). This filters out models susceptible to over-training. The best genomes from the validation set were then examined on the test set.

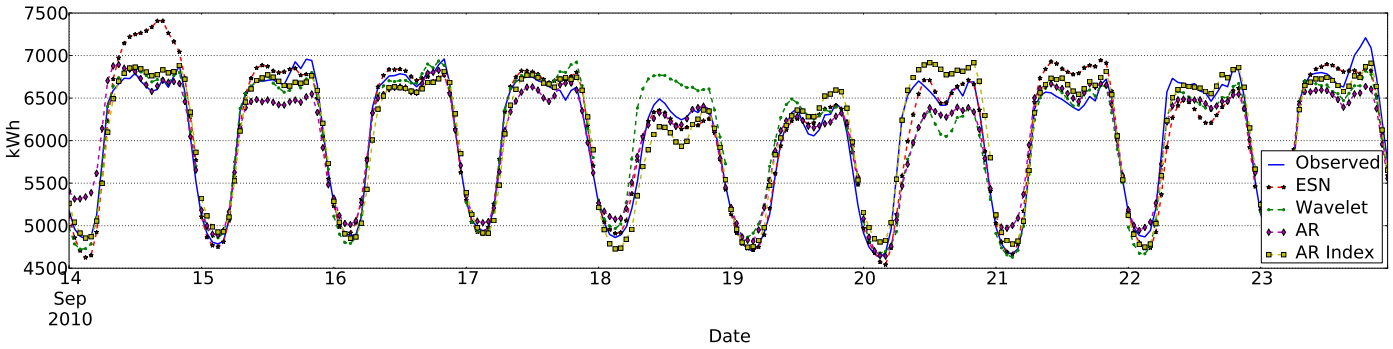
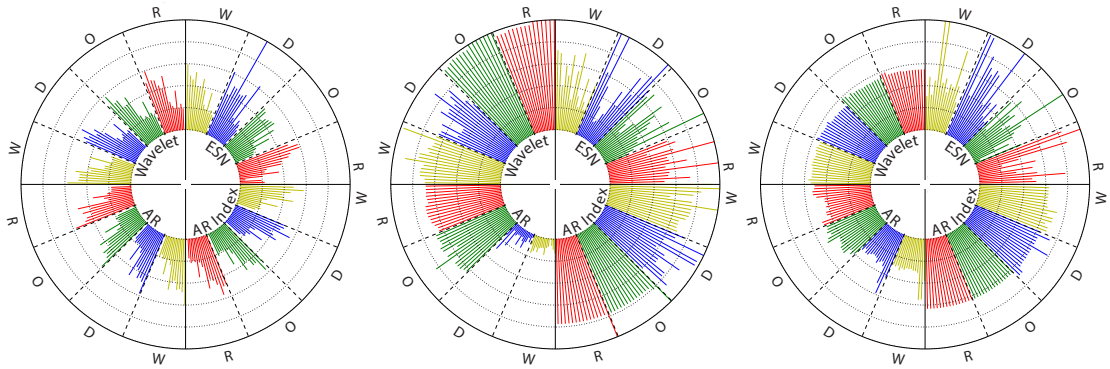


Fig. 3: A sample of 10 consecutive days of 24-hour predictions from the test set, as performed by the model with the lowest error on the validation set across all evolutionary runs.



(a) Single household (b) Distribution substation (c) Transmission

Fig. 4: Polar plots showing the prediction error on the test set (after validation) for the best cleansing-prediction combination found in each evolutionary run. Each quadrant shows the predictions for a given model. Each sector in turn indicates the cleansing method: R = raw (no cleansing), O = original algorithm [9], D = daily means subtracted, W = weekly means subtracted.

TABLE II: Test set prediction error of the best evolved predictor for each cleansing/prediction model/dataset combination. For each prediction model and dataset, the cleansing method giving the lowest error on the test set is indicated in bold.

	Single user				Distribution substation				Transmission			
	Raw	Original	Day	Week	Raw	Original	Day	Week	Raw	Original	Day	Week
ESN	0.308	0.111	**0.110**	0.113	20.72	**20.13**	21.49	20.86	296	306	292	**261**
Wavelet	0.083	**0.082**	0.084	0.084	22.34	22.33	**21.06**	21.40	316	316	**313**	314
AR	0.087	0.087	**0.084**	0.085	21.43	21.43	19.99	**19.86**	312	309	294	**290**
AR Index	0.083	0.083	0.083	**0.078**	21.68	21.68	**21.21**	21.45	**324**	325	331	330

In order to illustrate the efficiency of the evolved predictors, 10 consecutive 24-hour predictions on the transmission data are shown in Fig. 3. For each model, the prediction is done by the best individual as ranked by error on the validation set. In general, we find that all the models evolve reasonably precise predictors. We note that the ESN seems to cope better than the other models with the transitions between weekdays and weekends (Sep. 18 and Sep. 20 in Fig. 3). Fig. 4 further demonstrates the behavior of the different cleansing and prediction model combinations for each dataset. This figure shows the test set prediction error for the 15 (out of the original 30) genomes that performed best on the validation set in each experiment, with the length of each line indicating the prediction error on the test set for the best evolved model in the corresponding experiment. With regard to performance, we see a greater spread in the echo state networks than in the other models. This is partly due to the large number of parameters in the ESN genome, leading to a large search space, and partly to the topology of the network. Some of the ESNs had a tendency to get stuck in a destructive feedback loop when executed on the test set, resulting in very bad predictions. This was only partly filtered out by the validation process. Interestingly, this effect is most pronounced on the transmission dataset. We believe this is due to the high regularity of this dataset, allowing the ESN to overtrain, resulting in a higher to susceptibility to become unstable when faced with novel data.

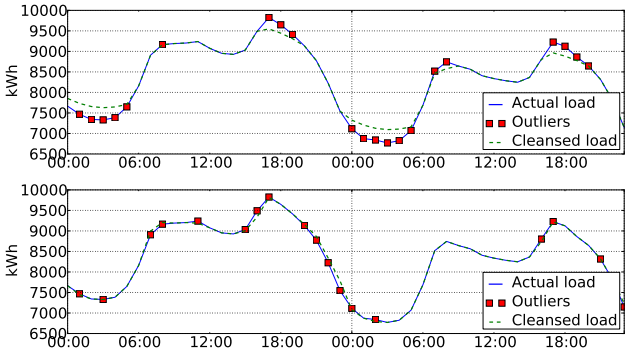
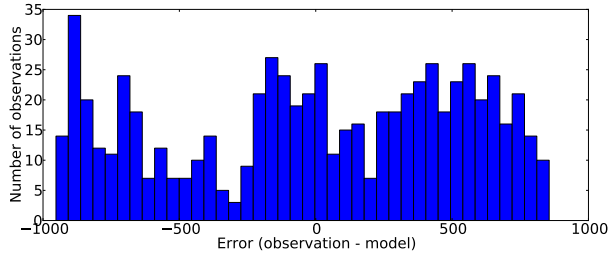
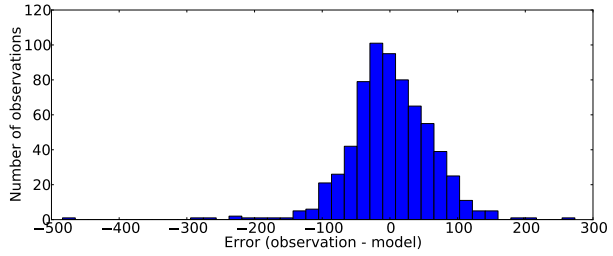


Fig. 5: Two days of cleansed load data. Top figure: using the original algorithm from Chen et al. [9]. Bottom figure: cleansed after subtracting typical weekly pattern. For illustration purposes, smoothing and Z-score parameters are chosen to identify a large number of outliers. The original algorithm has a higher tendency to cut peaks and fill troughs, which is undesirable.



(a) Original algorithm



(b) After removing typical week

Fig. 6: Distribution of "error" for a randomly chosen 4-week period, i.e. distance from raw data to smoothing spline: (a) using the original algorithm [9], and (b) when cleansed after subtracting typical weekly pattern. We note that the B-splines methodology implicitly assumes these errors to follow a Normal distribution, which seems more appropriate for the plot in part (b).

In the single household experiments, each set of 30 evolutionary runs was performed on the same set of randomly chosen households. In Fig. 4a, we see that the prediction error follows a similar pattern for all the cleansing/prediction model combinations, indicating that performance is mainly affected by the choice of household. This is not a surprising result, since the load profiles of each household are very different from one another.

Table II shows the test set prediction error for the genome with smallest validation set error for each combination of cleansing method, prediction model and dataset. For each combination of prediction model and dataset, the cleansing method resulting in lowest test set prediction error is indicated in bold. In 9 out of 12 cases, our proposed method of cleansing improves upon the predictions obtained when using the method by Chen et al. [9]. From Fig. 4, we see that our proposed cleansing method typically yields higher variance than without cleansing. This is as expected, since the four additional genes make the search space considerably more complex. The lack of variance seen when using the original cleansing method similarly indicates that this approach has little effect on the prediction performance. This is consistent with our original hypothesis.

To further examine the differences between the two approaches to load cleansing, see Fig. 5. As expected, the original algorithm tends to shave the peaks and fill the troughs, whereas the transformation suggested in this paper leads to a more uniform distribution of outliers.

Finally, Fig. 6 shows the smoothing "error" for the different cleansing methods, i.e. the difference between the spline and the observed data, see Eq. (2). We see that our transformation yields errors that follow a Normal distribution to a larger degree than the original. This is in accordance with our hypothesis and consistent with the rest of the cleansing framework. Moreover, the assumption of independence is more valid in the approach proposed here than in the original method. We calculated autocorrelations for a randomly chosen 4-week period at 1-hour and 24-hour lags for the two cleansing methods, and observed a reduction from 0.87 to 0.50 (lag 1) and from 0.93 to 0.21 (24 hours). Removal of the daily pattern thus reduces autocorrelation at key lags significantly, indicating a dataset better suited for B-splines smoothing.

IV. Conclusion

We have considered short-term load prediction for a group of 150 residential users from a small city in southern Norway, as well as transmission level load prediction on a dataset from British Columbia. We argue that data cleansing and prediction should be performed as a combined endeavor, where the accuracy of the prediction can be used to gauge the efficiency of the cleansing effort.

We observe that the typical pattern exhibited by power load time series tend to violate the assumption of normally and independently distributed noise made by several load cleansing approaches. In order to mitigate this effect, we have proposed two ways to transform the original signal such that the above assumption is more correct. Our experimental results indicate that these two methods of eliminating typical load patterns improves the perfomance of the load prediction algorithms,

making load cleansing contribute to the overall performance of the load prediction system.

The high variance in test set prediction error for the ESN indicates that this model suffers from instability, but also that it has potential to produce very accurate predictors. Different evolutionary strategies could consistently produce better performing genomes and help avoid the unstable phenotypes.

ACKNOWLEDGMENTS

This work has been conducted as part of the project "Next Generation Control Centres for Smart Grids" run through the Norwegian Smart Grid Centre. We thank Nicolai Feilberg at SINTEF Energy Research for helping us to to prepare and understand the dataset.

REFERENCES

[1] D.W. Bunn and E.D. Farmer. Review of short-term forecasting methods in the electric power industry. In *Comparative Models for Electrical Load Forecasting*, pages 13–30. Wiley, 1985.

[2] H. K. Alfares and M. Nazeeruddin. Electric load forecasting: literature survey and classification of methods. *International Journal of Systems Science*, 33(1):23–34, 2002.

[3] A. Khosravi, S. Nahavandi, and D. Creighton. Construction of optimal prediction intervals for load forecasting problems. *Power Systems, IEEE Transactions on*, 25(3):1496–1503, 2010.

[4] N. Amjady. Short-term hourly load forecasting using time-series modeling with peak load estimation capability. *Power Systems, IEEE Transactions on*, 16(3):498–505, 2001.

[5] Hong-Tzer Yang, Chao-Ming Huang, and Ching-Lien Huang. Identification of armax model for short term load forecasting: an evolutionary programming approach. In *Power Industry Computer Application Conference, 1995. Conference Proceedings., 1995 IEEE*, pages 325 – 330, 1995.

[6] Kwang-Ho Kim, Jong-Keun Park, Kab-Ju Hwang, and Sung-Hak Kim. Implementation of hybrid short-term load forecasting system using artificial neural networks and fuzzy expert systems. *Power Systems, IEEE Transactions on*, 10(3):1534–1539, 1995.

[7] D. Srinivasan, Swee Sien Tan, C.S. Cheng, and Eng Kiat Chan. Parallel neural network-fuzzy expert system strategy for short-term load forecasting: system implementation and performance evaluation. *Power Systems, IEEE Transactions on*, 14(3):1100 –1106, 1999.

[8] N. Amjady, F. Keynia, and H. Zareipour. Short-term load forecast of microgrids by a new bilevel prediction strategy. *Smart Grid, IEEE Transactions on*, 1(3):286–294, 2010.

[9] Jiyi Chen, Wenyuan Li, A. Lau, Jiguo Cao, and Ke Wang. Automated load curve data cleansing in power systems. *Smart Grid, IEEE Transactions on*, 1(2):213–221, 2010.

[10] Agoston E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer Verlag, 2003.

[11] Andrei Z. Morch, Ingeborg Graabak, and Nicolai Feilberg. Results of Monitoring of AMR Systems in Norway: Analysis of Metered Data and Definition of the Performance Parameters. In *20th International Conference on Electricity Distribution*, 2009.

[12] Hanne Marit Dalen and Bodil Merethe Larsen. Formålsfordeling av husholdningenes elektrisitetsforbruk i 2006. Technical report, Statistisk sentralbyrå, 2009.

[13] T. Hastie, R.J. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd Edition)*. Springer series in statistics. Springer-Verlag New York, 2 edition, 2009.

[14] J. O. Ramsay and B.W. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, 2 edition, 2005.

[15] G. Mateos and G. B. Giannakis. Robust nonparametric regression via sparsity control with application to load curve data cleansing. *IEEE Transactions on Signal Processing*, 60(4):1571–1584, 2012.

[16] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[17] Mark D. Skowronski and John G. Harris. Automatic speech recognition using a predictive echo state network classifier. *Neural Networks*, 20(3):414–423, 2007.

[18] Š. Babinec and J. Pospíchal. Optimization of echo state neural networks for electric load forecasting. *Neural Network World*, 2(7):133–152, 2007.

[19] H. Showkati, A.H. Hejazi, and S. Elyasi. Short term load forecasting using echo state networks. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–5, 2010.

[20] H. Showkati, A.H. Hejazi, and S. Elyasi. Short term load forecasting using echo state networks. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–5, 2010.

[21] Herbert Jaeger, Mantas Lukoševičius, Dan Popovici, and Udo Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007.

[22] V. Dordonnat, S.J. Koopman, M. Ooms, A. Dessertaine, and J. Collet. An hourly periodic state space model for modelling French national electricity load. *International Journal of Forecasting*, 24(4):566–587, 2008.

[23] O. Renaud, J.L. Starck, and F. Murtagh. Wavelet-based combined signal filtering and prediction. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(6):1241–1251, 2005.

[24] D. Benaouda, F. Murtagh, J.L. Starck, and O. Renaud. Wavelet-based nonlinear multiscale decomposition model for electricity load forecasting. *Neurocomputing*, 70(1):139–154, 2006.