Very Short-Term Load Forecasting: Wavelet Neural Networks With Data Pre-Filtering

Che Guan, *Student Member, IEEE*, Peter B. Luh, *Fellow, IEEE*, Laurent D. Michel, Yuting Wang, and Peter B. Friedland, *Member, IEEE*

***Abstract*—Very short-term load forecasting predicts the loads 1 h into the future in 5-min steps in a moving window manner based on real-time data collected. Effective forecasting is important in area generation control and resource dispatch. It is however difficult in view of the noisy data collection process and complicated load features. This paper presents a method of wavelet neural networks with data pre-filtering. The key idea is to use a spike filtering technique to detect spikes in load data and correct them. Wavelet decomposition is then used to decompose the filtered loads into multiple components at different frequencies, separate neural networks are applied to capture the features of individual components, and results of neural networks are then combined to form the final forecasts. To perform moving forecasts, 12 dedicated wavelet neural networks are used based on test results. Numerical testing demonstrates the effects of data pre-filtering and the accuracy of wavelet neural networks based on a data set from ISO New England.**

***Index Terms*—Neural networks, pre-filtering, very short-term load forecasting, wavelet and filter bank.**

I. Introduction

VERY short-term load forecasting (VSTLF) predicts the loads one or several hours into the future in steps of a few minutes (e.g., 5 min) in a moving window manner based on online data collected every few seconds (e.g., 4 s). Accurate load forecasting has traditionally been important since it is critical for automatic generation control and resource dispatch, and it also ensures revenue adequacy for the independent system operator (ISO) multi-settlement markets. Effective VSTLF, however, is difficult in view of the noisy data collection process with possible malfunctioning of data gathering devices and complicated load features.

Methods for very short-term load forecasting are limited. Existing methods of persistence, extrapolation, time series, Kalman filtering, fuzzy logic, and neural networks (NN) will be reviewed in Section II. Among these methods, neural networks have been widely used. A standard NN was used for VSTLF in [2]. To improve data stationarity, inputs to an NN were transformed by using logarithmic differences in [3] and by using relative increments in [4]. A single neural network, however, may not be able to accurately capture complicated load features because the load data have multiple frequency components, and each may have its unique pattern. Furthermore, spikes are randomly distributed over time and have different magnitudes and widths. They affect neural network training, and result in degraded predictions. An intuitive way to filter the spike is to compare the measured and predicted loads, and if the absolute value of the difference is greater than a threshold, a spike is said to be detected. The spike was then replaced by the interpolated value [3] or the predicted value [5]. This way, however, may not be effective. To reduce the effects of spikes, further analysis and filtering are needed.

Recently, we have developed a method for short-term load forecasting (STLF) which predicts the loads of tomorrow in hourly steps based on the single-level wavelet decomposition and neural networks trained through using a data set from ISO New England [6]. A correction coefficient scheme was also developed to enhance predictions around holidays [7]. These methods presented a way for handling load features at different frequencies. However, the load features of STLF are quite different from the ones of VSTLF because short-term load data have fewer patterns than very short-term load data to be analyzed in Section IV-A. Also, spikes were not considered because they had been removed by ISO New England before STLF was performed, whereas removing spikes is a critical issue for VSTLF.

In this paper, wavelet neural networks (WNN) with data pre-filtering are developed to forecast the loads 1 h into the future in 5-min steps in a moving window manner. To effectively remove spikes, it is observed that spikes may have different magnitudes and widths. Thus, they are classified into micro and macro spikes at either 4-s or 5-min resolutions. Micro and macro filtering techniques are developed in Section III to effectively detect and filter them out. The advantage of filtering spikes in the 4-s data series is to provide the leading indicator to the operator of a potential SCADA telemetry problem in real-time. Filtering spikes in the 5-min data series is often a lagging indicator of faulty load telemetry.

Wavelet neural networks are developed in Section IV. The wavelet technique is used to decompose the loads into multiple frequency components. Each component is then appropriately transformed, normalized, and fed with time and date indices to a neural network, so that the features of individual

Manuscript received February 11, 2011; revised July 26, 2011, December 29, 2011, March 25, 2012, and April 11, 2012; accepted April 16, 2012. Date of publication June 05, 2012; date of current version January 17, 2013. This work was supported in part by ISO New England. A preliminary version of this paper was presented at IEEE PES General Meeting 2009 [1]. Paper no. TPWRS-00119-2011.

C. Guan and P. B. Luh are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-2157 USA.

L. D. Michel and Y. Wang are with the Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269-2155 USA.

P. B. Friedland is with ISO New England, Holyoke, MA 01040 USA.

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

Digital Object Identifier 10.1109/TPWRS.2012.2197639

components are properly captured. Forecasts from individual neural networks are then transformed back and combined to form the final forecasts. To perform moving forecasts, 12 dedicated wavelet neural networks are used based on test results.

In Section V, the method is configured through training, validation, and test data sets as presented in [8, Ch. 2]. Example 1 uses a classroom-type problem to illustrate the effects of the wavelet decomposition. Based on the data set from ISO New England (ISO-NE), Example 2 demonstrates the values of data pre-filtering, wavelet decomposition, load transformation, neural networks, and dedicated wavelet neural networks for VSTLF. The code as well as part of the test data and results are open, and can be downloaded at http://github.com/ldm-bouge/vstlf.

A preliminary version of this paper was presented in [1] where spikes were classified and filtered. A single WNN was then established for VSTLF. Based on this preliminary result, we analyze the property of spikes with respect to magnitudes and widths, and discuss spike filtering methods and their parameters. After a thorough data analysis, additional NN inputs are tried and selected. The parameters of the filter bank in wavelet transform are then discussed, derived, and selected. Finally, 12 dedicated WNNs are developed to perform moving forecasts. For our method, model parameters are determined through training, validation, and test processes in a three-way data split.

II. Literature Review

Not many papers report the handling of spikes. One way is to compare measured and predicted loads, and if the absolute values of the differences are greater than a threshold, spikes are declared and then replaced by predicted values in [5]. Another way is to replace observed spikes by zeros which are then fixed by using a splining algorithm. If the length of zeros is too long, interpolations from a similar day's loads are used to fill zero-valued data [3]. These methods are valuable. However, they are prone to errors due to the uncertain nature of the load data and the various magnitudes and widths of spikes. Spikes replaced by bad values may degrade future predictions. Therefore, spikes have to be further analyzed, and effective ways are highly needed for filtering them out.

Spike filtering has also been reported for short-term load forecasting. In comparison to VSTLF, spikes in STLF have different features with respect to magnitudes and widths because of the integrative nature of short-term load data and the fact that most spikes should have been removed before STLF is performed. The simple techniques consisting of if-then rules, low pass filtering, and NN based self-filtering were used to handle STLF spikes in [9]. Recently, entropy related functions, which are robust to noisy data, were developed in [10] and were further applied to the training of neural networks for future three-day wind power forecasting [11]. To perform the online training, a self-adaptive approach was used in [11] where "the information potential of the error" was recursively estimated. Although these methods are robust to noisy data, in order to help a forecasting model learn normal load patterns rather than complicated noisy data in real-time, it is desirable to remove spikes before data are used for VSTLF.

Limited VSTLF methods have been reported in the literature, and they include methods of persistence, extrapolation, time series, fuzzy logic, Kalman filtering, and neural networks. Persistence forecasting [12] may be the simplest method, and it assumes that the forecast data will be the same as the last measured values. This is not sufficient for VSTLF because very short-term load series change in real-time. Extrapolation predicts the load based on the past by using a least square algorithm [13] or by using a curve fitting algorithm based on a shape similarity criterion [14]. The load increment was predicted through a weighted average of increments of previous loads in [15]. A dynamic clustering method was used to pre-group the loads into multiple groups, and load increments were then forecasted in [16].

Similar to the extrapolation method, the auto-regression method uses a simple linear combination of the previous load series for prediction(s). Its coefficients were tuned online using the least mean square algorithm in [2]. The method was extended to autoregressive integrated moving average (ARIMA) for load forecasting, and parameters were updated via a recursive least square algorithm with a forgetting factor in [17]. ARIMA was extended to seasonal autoregressive integrated moving average to capture the seasonal load feature in [18]. Support vector regression method was developed for VSTLF, which was used with kernel functions to create complex nonlinear decision boundaries in [19]. Holt-Winters adaptation and the new intraday cycle exponential smoothing method were used together for predictions in [20].

Kalman filter was applied to VSTLF in a few references. For example, the loads were separated into deterministic and stochastic components, and both were predicted via Kalman filters in [21]. While in [5], the deterministic and stochastic components were predicted via the least square algorithm and Kalman filter, respectively. Fuzzy logic methods convert input data to fuzzy values which are then compared with patterns extracted from the training process. The most similar fuzzy value was chosen and then mapped to the prediction in [2]. Fuzzy logic was also combined with neural networks to form a fuzzy neuron system, and the parameters of which were configured via chaotic dynamics reconstruction techniques in [22] and [23]. A hybrid neuron-fuzzy approach was developed in [24] which used the cross validation methodology to choose inputs, membership functions, and optimization methods.

Among all these VSTLF methods, neural networks have been widely used. They assume a nonlinear functional relationship between the loads to be forecasted and affecting factors, and estimate the weights based on historical data. Their inputs may include the time and date indices, the loads of previous hour, and the loads of yesterday and last week with the same time and date indices to the forecasting hour. For example, different feature sets of historical load data were tested in [25]. Weather information is seldom used for VSTLF due to the large time constant of the load [4]. Transformations of load inputs, e.g., the logarithmic difference and relative increment, have been reported to improve data stationarity in [3] and [4]. Also, different neural networks were used for different periods of a day [4]. These neural network methods provide valuable information for the input selection and transformation. However, very short-term load data have complicated features, and few papers present thorough analysis and effective ways to capture load features in real-time.

III. Data Pre-Filtering

For ISO New England, load data are collected from data collecting devices every 4 s and then aggregated into 5-min loads. Because of possible malfunctioning of collecting devices, spikes exist within load data. These spikes do not reflect true loads and, as a result, affect NN training and degrade predictions. Potential spikes are observed having varying magnitudes and widths at either 4-s or 5-min resolutions, and they are randomly distributed over time. A spike is said to be detected if the absolute value of the difference between the original load and this smoothed load exceeds a threshold. Spikes are then classified as "micro spikes" and "macro spikes" based on their widths. A micro spike is defined if its width is smaller than a threshold w_1 (in terms of number of resolution units of either 4 s or 5 min), whereas a macro spike is defined if its width is in-between two thresholds w_1 and w_2. These thresholds are determined based on training, validation, and test data sets. It is difficult to differentiate the spikes with widths larger than w_2 from regular load changes. However, this situation usually requires human intervention, and will not be considered.

To filter these spikes, micro spikes are first recognized at the 4-s resolution and are removed by using the micro spike filtering method which will be presented in Section III-A. After aggregating into 5-min loads, micro spikes are again recognized and filtered at the 5-min resolution by using the same method. Finally, macro spikes are recognized and processed at the 5-min resolution by using the macro spike filtering method which will be presented in Section III-B. Macro spike filtering is only applied to the loads at the 5-min resolution because macro spikes at the 4-s resolution may become micro spikes after integration.

A. Micro Spike Filtering

The key idea for filtering the micro spike is the use of a zero phase filter to obtain the smoothed load. If the absolute value of the difference between the original load and this smoothed load exceeds a threshold, a spike is said to be detected. Then, the spike is replaced by the smoothed load. This method is first applied to the loads at the 4-s time resolution and then at the 5-min time resolution.

Intuitively, the response of a zero phase filter to a rectangular pulse function should be a smoothed and symmetric function without shifting in time. This filter is realized by a unit impulse response symmetric with respect to the time zero axis. When taking Fourier transform, the resulting function should have the phase identically equal to zero. Such a filter is called a zero phase filter as described in [26, Ch. 19]. In practice, the idea is to take the average of the actual data in the time-forward and reversed operations with equal weights over the filter window as explained below [27, pp. 604–605]. The result from the zero phase filter has precisely zero phase distortion and magnitude modified. Let the input sequence at time $t+N$ be denoted as $X=\{x(t+1),\ldots,x(t+N)\}$, where N is the length of the latest load inputs to be processed in real-time. Sequence $Y=\{y(t+w),\ldots,y(t+N)\}$ is sequentially produced by the filter with the width w in the following time-forward operation of the zero phase filter:

$$y(t+n)=\sum_{i=n-w+1}^{n}\frac{x(t+i)}{w},\quad n=w,\ldots,N. \tag{1}$$

The above sequence is appended by $\{x(t+N+1),\ldots,x(t+N+w-1)\}$ (explained in the next paragraph) to make sure the sequence $\{y(t+N-w+1),\ldots,y(t+N)\}$ after the time-reversed operation of the zero phase filter has a similar magnitude to the load segment $\{y(t+w),\ldots,y(t+N-w)\}$. Following [27], the resulting sequence is reversed and run through the same filter again. The output of this second filtering is then time reversed to generate the final smoothed series $Z=\{z(t+w),\ldots,z(t+N)\}$.

Since a zero phase filter is causal, the load inputs have to be appended. The load segment $\{x(t+N+1),\ldots,x(t+N+w-1)\}$ is available during training. However, it is not available during real-time forecasting. To append load inputs with a reasonable sequence, the load segment $\{x(t+N-w+1),\ldots,x(t+N)\}$ is mirrored horizontally and flipped vertically with respect to the point $(t+N,x(t+N))$ in the coordinate space. Based on observation, this is because the changes of load series over a short time period have similar slopes for most of the times.

To detect spikes, a sequence of the difference $D=\{d(t+w),\ldots,d(t+N)\}$ between the smoothed and actual is obtained by

$$d(t+n)=z(t+n)-x(t+n),\quad n=w,\ldots,N. \tag{2}$$

A micro spike is said to be detected if the absolute value of $d(t+n)$ exceeds a threshold m, and the width of the spike is smaller than a threshold w_1 (the width of the processing window). To replace a spike with a corrected signal, the value of $x(t+n)$ is replaced with $z(t+n)$. These thresholds are analyzed and then determined through training, validation, and test processes in a three-way data split.

Fig. 1 depicts the 4-s load series before and after the micro spike filter is applied. The spikes with widths smaller than the processing window w_1 (micro spikes, as marked by the three small red circles) are removed by the filter. Spikes with widths close to or greater than the processing window w_1 (macro spikes, as marked by the large black ellipse) are only attenuated or cannot be handled by the micro spike filter at the 4-s time resolution. However, they may become micro spikes after integration, and can then be handled by the same method at the 5-min resolution. In this way, all micro spikes within the processing window are detected and replaced by smoothed loads, whereas the load data outside the window are not touched.

B. Macro Spike Filtering

The key idea for filtering out macro spikes is to detect a pair of edges, and fix the loads between the two edges with linear interpolation values. This method is only applied to the loads at the 5-min resolution because macro spikes at the 4-s resolution may become micro spikes after integration. To detect edges, the first-order differencing transformation is applied to the load series at the 5-min resolution. The edge is said to be detected when

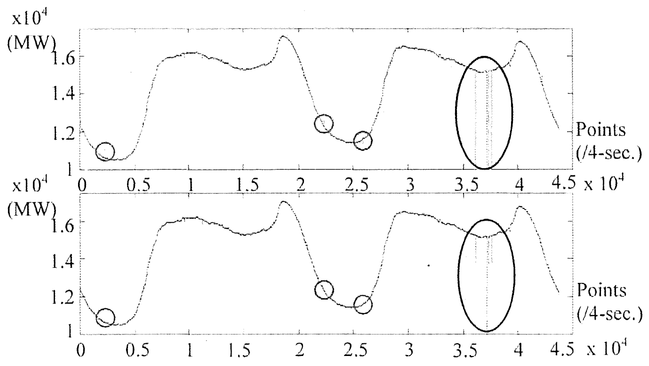


Fig. 1. Before (top) and after (bottom) micro and macro spike filtering based on two days of continuous ISO-NE load data at the 4-s resolution.

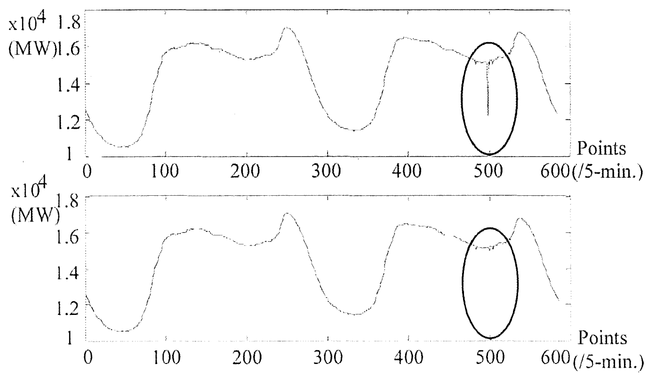


Fig. 2. Before (top) and after (bottom) macro spike filtering at the 5-min resolution.

the absolute value of the difference exceeds the threshold m. A macro spike is then said to be recognized when two sequential edges are located, and the width of the two edges is less than a threshold w_2 and equal to or greater than the threshold w_1. The spike whose width is less than w_1 is a micro spike, and should have been removed in micro spike filtering which is described in Section II-A. To fix a macro spike, the load in-between the two edges is replaced by a value from linear interpolation. This interpolation method is used because the changes of 5-min loads over a short time period have similar slopes for most of the times based on observation. Fig. 2 depicts the 5-min load series (4-s integrated into 5-min loads depicted in the second plot of Fig. 1) before and after the macro filtering is applied.

IV. Wavelet Neural Networks

To perform accurate predictions after pre-filtering, load properties are analyzed in Section IV-A. Data analysis shows that the load data have different components: a very fast changing component from five to 15-min resolutions, a fast changing component from 15-min to 1-h resolutions, and a slow changing component with hourly, weekly, and monthly patterns. The WNN method is developed to capture the complicated load properties. To accurately capture load features at multiple frequencies, a wavelet technique is used to decompose the loads into several frequency components in Section IV-B. Due to the use of convolution in the wavelet transform, additional data need to be

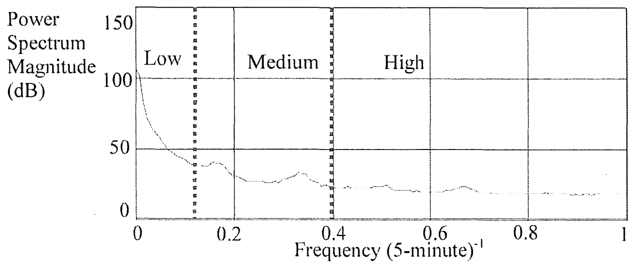


Fig. 3. Power spectrum density for 5-min load data (January 1, 2007 to June 30, 2008).

padded at the end side of the load segment in real-time. Relationships among the padding parameters are discussed and derived. Different padding strategies are then tested, and the best one is determined via the test data set. In Section IV-C, each load component is properly transformed and then fed with other time and date indices to a separate neural network. Predictions from individual neural networks are combined to form the forecasts. Finally, 12 dedicated wavelet neural networks are used to perform moving forecasts in Section IV-D.

A. Load Property Analysis

Very short-term load data have complicated properties. They are illustrated by the power spectrum density which describes how the power of load data is distributed with frequency. As shown in Fig. 3, the main power lies in the low frequency and several small pedals afterward, and each one has a unique frequency component. Intuitively, this frequency domain is divided into three frequency components as denoted by low, medium, and high frequencies. If each one is further magnified by amplitude spectrum (explained in the next paragraph), it is observed that these components have different features.

As depicted in Fig. 4, the amplitude spectrum shows that the low, medium, and high load frequency components have different features. Spectral lines for the low frequency component in the eclipse are magnified further. These spectral lines represent unique load patterns, and the ones located at frequencies corresponding to hourly, weekly, and monthly information are marked. The amplitude spectrums for the medium and high load frequencies (reflecting fast changes in load data) have small magnitudes, and hence are not magnified. Dashed lines are used to separate load components as they are in the separation in Fig. 3.

B. Filter Bank in Wavelet Transform

The load data have multiple frequency components as depicted in Fig. 3, and each may have a unique pattern as depicted in Fig. 4. An intuitive idea is to decompose the loads into multiple frequency components and process each independently. For example, the load data were decomposed into multiple resolution scales in [28] and [29]. Fourier transform is a straightforward technique to represent the signal as a sum of sinusoids which are only localized in frequency. In contrast to Fourier transform, wavelets are localized in both time and frequency and often give a better representation using multi-resolution analysis. A detailed introduction to wavelets can be found in [30,

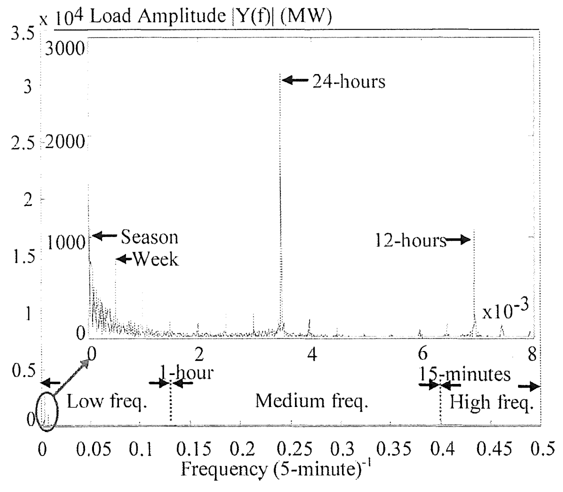


Fig. 4. Amplitude spectrum for 5-min load data.

Ch. 1]. Motivated by the successful one-level wavelet decomposition for short-term load forecasting in our previous work [6], a wavelet technique is chosen to decompose input loads into multiple frequency components. The input loads are first decomposed into low (L) and high (H) frequency components at level one. The L frequency component called "approximation" represents a general trend of the signal, whereas the H frequency component is viewed as a difference between two successive approximations [28]. Since the load has a large magnitude and multiple frequency information, the L frequency component is further decomposed into low-low (LL) and low-high (LH) frequency components. There is no need to decompose the H component because it has a small magnitude as compared to the L frequency component. The decomposed level is analyzed later on. Components LL, LH, and H are very similar to the low, medium, and high frequencies described in Section IV-A.

To implement the two-level wavelet transform, a three-channel filter bank is used as shown in Fig. 5. The high frequency channel consists of the analysis and synthesis stages. At the analysis stage, a high pass filter (a wavelet function that plays the role of the anti-alising) G_1 filters out the low frequency component. A down-sampling step then removes the odd-numbered data points. At the synthesis stage, the up-sampling step pads zeros to down-sampled data to recover the data length. A high pass filter H_1 then removes the replicas of signal spectrum caused by up-sampling. Similarly, the low-high frequency channel uses a low pass filter G_0 to compute the general trend, and then holds the even-numbered points. Next, these points are further decomposed into two parts. The low-high part convolves with G_1 and then takes steps similar to those for the high frequency channel. To recover the initial input length, the output from H_1 has to be up-sampled and convolve with H_0. These are the steps to produce the LH frequency component. The same is true for the low-low frequency channel. Filters G_0, G_1, H_0, and H_1 have to satisfy perfect reconstruction and orthogonality [31].

The filter bank in the wavelet transform described above adopts a circular convolution as explained in [31, Ch. 8]. Circular convolution causes boundary distortions which affect

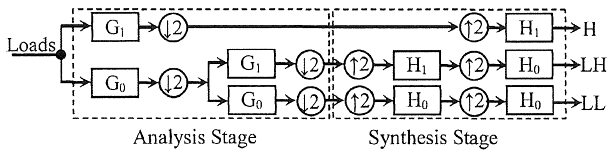


Fig. 5. Three-channel filter bank.

neural network predictions. To reduce the distortion, it is necessary to extend the signal beyond the boundaries. In the high frequency channel shown in Fig. 5, the distortion length for a convolution between the input loads and G_1 is $(lw - 1)$ based on the convolution theory, where lw is the filter length. Down-sampling and up-sampling do not produce the distortion. H_1 introduces another distortion with the same length $(lw - 1)$. The total distortion length is $2(lw - 1)$. The low-high frequency channel sequentially convolves the inputs with four filters (G_0, G_1, H_1, and H_0) with a final length of the distortion $4(lw - 1)$, doubling that of the high frequency channel. The same is true for the low-low frequency channel. The distortion length is thus roughly doubled for a component which is further decomposed one more level. A detailed analysis can be found in [32]. To make sure that at least one value is not affected by distortion, the load inputs to NN need to be padded. The padding length has to be equal to or greater than the distortion length (wmaxlev function in MATLAB Wavelet Toolbox):

$$lx = (lw - 1) \cdot 2^{lvl} \tag{3}$$

where lx is the distortion length which indicates the minimum padding length, and lvl is the level of the decomposition. Hence, the total length for load inputs to be decomposed has to be equal to or greater than the sum of the minimum padding length lx and the length of the load inputs of the last hour (12 points). For VSTLF, the latest historical data are available and used to pad the last hour's loads at the front. Additional data are needed to pad the last hour's loads at the end, as discussed in the end of this subsection.

From (3), the relationships among the decomposition level lvl, the filter (G and H) length lw, and the minimum padding length lx are very close. It can be concluded that fixing lvl and increasing lw, or vice versa, will increase lx. This indicates that the padding length will increase, which may not be good because a long padding to load inputs can result in a poor training and prediction for NN. However, lvl should not be too small because the features of load components cannot be fully captured. The same is true for lw because a small lw has a poor ability to represent the load component behaviors. It is clear that neither lw nor lvl should be too large or small, so that a reasonable lx can be obtained. Therefore, a balance among lvl, lw, and lx has to be made due to their close relationships in (3).

To choose a good lvl for decomposition, different values are tested and compared, while lw and lx remain fixed. Two-level decomposition is found to be the best among levels from zero to three in Example 2 in Section V. This corresponds to the scheme presented in Fig. 5 with three decomposed frequency components H, LH, and LL. To choose a good filter length lw, a proper wavelet has to be chosen using the previous fixed lx and newly determined $lvl(= 2)$. Daubechies (Db) wavelets

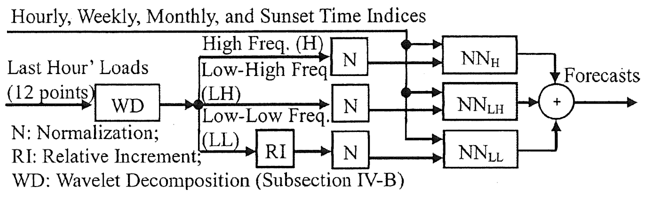

Hourly, Weekly, Monthly, and Sunset Time Indices
Last Hour' Loads (12 points)
WD
High Freq. (H)
Low-High Freq (LH)
Low-Low Freq. (LL)
N
RI
NN_H
NN_LH
NN_LL
+
Forecasts
N: Normalization;
RI: Relative Increment;
WD: Wavelet Decomposition (Subsection IV-B)

Fig. 6. Structure of wavelet neural networks.

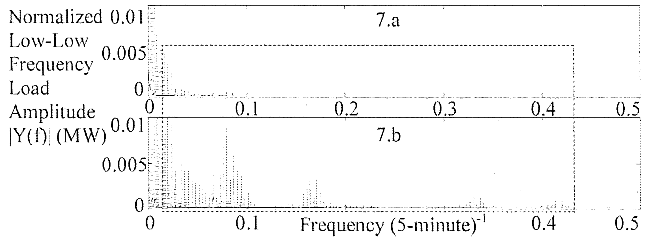


Fig. 7. (a) Amplitude spectrum for normalized low-low frequency before applying RI. (b) Amplitude spectrum for normalized low-low frequency after applying RI.

are adopted in our method because they belong to a family of orthogonal wavelets and are characterized by frequency responses having maximum flatness (at 0 and π). Db members tested are Db2-Db20 (even index numbers only). The index number refers to the filter length lw, and has the ability to represent complicated behaviors of signal components. For example, Db2 encodes constant components, and Db4 encodes constant and linear ones. However, the Db number cannot be too large. Otherwise, the minimum padding length will increase. Based on observation, the changes of load series over a short time period have similar slopes for most of the times. Hence, Db4 seems to be a reasonable choice because it encodes linear signal components, and is demonstrated to be the best among all the index numbers tested as presented in Section V.

Once $lvl(=2)$ and $lw(=4)$ are fixed in (3), lx can be calculated ($(lw-1)\cdot 2^{lvl}=12$). Since the last hour's loads (12 points) are used as NN inputs, the total length for load inputs to be decomposed has to be equal to or greater than the sum of the minimum padding length and the length of the last hour's loads (i.e., the total length ≥ 24). A more precise number can be calculated from the derivation in [32]. To further reduce distortion effects, padding strategies (e.g., zero-padding, periodic extension, and symmetrization) are tested. According to the test in Example 2 in Section V, symmetrization, a boundary replication which pads the loads by adding points symmetric to the original, is demonstrated to be the best strategy. This also corresponds with the conclusion on [31, p. 263]. These parameters are determined through training, validation, and test processes in a three-way data split.

C. Neural Networks

To capture decomposed frequencies, our idea is to properly transform individual components as presented in Section IV-A. The transformed components are then fed to separate neural networks. Finally, individual predictions from NNs are added to form the forecasts as depicted in Fig. 6.

The load components are treated differently. The LL frequency represents the majority of load information, including hourly, weekly, and monthly patterns as analyzed in Section IV-A. Since the loads from 5- to 60-min outs are predicted each time, the loads of the last hour (lag $=12$) are used as inputs. Loads with other lags are also tested, but the results are not further improved. To remove a first-order trend and anchor the predictions by the latest load, the relative increment (RI) in loads in [4] is applied:

$$LL_d^{RI}(t)=\frac{(LL_d(t)-LL_d(t-1))}{LL_d(t-1)} \tag{4}$$

where LL represents the low-low frequency load component at day index d, and t is the time index in a 5-min period. RI indicates the relative increment transformation and is used to stationarize the load component series. This transformation reveals more of the hidden information in the LL frequency component in Fig. 7(b) than the one without applying RI in Fig. 7(a). But the other observation shows that RI reveals less of hidden information in the LH and H components than the one without applying RI. Hence, RI transformation is only applied to the LL load component. All the components then have to be normalized and fed to individual NNs.

In addition to load inputs (5 to 60 min), time and date indices are parts of the neural network inputs, including hourly, weekly, and monthly indices. Furthermore, sunset time is included to capture the load feature related to the street lighting. These indices are used to help NNs indentify the periodical patterns of load data. Similarly, low-high and high frequency NNs adopt the same time and date indices but use the load components without RI transformation. Finally, results from three NNs are summed up to form final forecasts. Other additional inputs were tested but not considered because the results were not significantly improved. These inputs include: area control errors, frequencies, and some selected loads from history (e.g., loads of the last several hours, loads of selected hours from yesterday, similar day's loads, and so on). Based on the literature review, actual weather data and weather forecasts from related methods, e.g., the climatology method in [33], are seldom used for VSTLF inputs because of the large time constant of the load and weather relationship. Also, real-time weather data are not available from ISO New England.

To narrow the numerous choices of input candidates down, different combinations of data inputs are screened based on small data sets. For example, load data from November 2007 to December 2007 are used for training, and loads for January 2008 are then predicted. The resulting candidate inputs are then examined through training, validation, and test processes in a three-way data split.

D. Moving Forecasts

When performing moving forecasts every 5 min, the intuitive approach would be to train a single WNN offline with historical data as presented in [34, Ch. 4] and train the WNN online whenever a new data point is available. This is the same as the self-adaptive training process of [35]. However, test results using this approach are not satisfactory. Based on further testing, our final configuration consists of 12 dedicated WNNs, one for each 5-min period in the hour. In this way, individual WNNs can be properly trained. For example, at 2:55 am, WNN_1 predicts the loads from 3:00 am to 3:55 am in 5-min periods; WNN_2 at time 3:00 am predicts the loads from 3:05 am to 4:00 am in 5-min periods, etc. Then at 3:55 am, when the measured values are known for the past 12 time steps, WNN_1 is trained online (updated) with the data from 3:00 am to 3:55 am and then predicts the loads from 4:00 am to 4:55 am, and the process repeats.

V. Numerical Test Results

The method was developed in MATLAB for prototype implementation and then converted to JAVA using Eclipse. The open source can be downloaded at http://github.com/ldmbouge/vstlf. In this section, the software was run on a server with dual Xeon quad core Intel E5620 2.4-GHz processors and 36 GB of memory. The performance measures include mean absolute error (MAE), mean absolute scaled error (MASE) as presented in [36] and [37], mean average percentage error (MAPE), and standard deviation of sample errors (SD):

$$MAE(k) = n^{-1} \sum_{t=k}^{12n+k} |L_p(t) - L_A(t)|$$

$$k = 1, \ldots, 12, \tag{5}$$

$$MASE(k) = \frac{MAE(k)_{out\text{-}of\text{-}sample}}{MAE_{in\text{-}sample}}, \tag{6}$$

$$MAPE(k) = n^{-1} \sum_{t=k}^{12n+k} \left(\frac{|L_p(t) - L_A(t)|}{L_A(t)} \right) \times 100\%, \tag{7}$$

$$SD(k) = \left[n^{-1} \sum_{t=k}^{12n+k} \left(L_p(t) - n^{-1} \sum_{t=k}^{12n+k} (L_p(t) - L_A(t)) \right) \right]^{1/2}. \tag{8}$$

In the above equations, index k represents 5 to 60 min in 5-min steps, n indicates the number of hours in the forecasting horizon, and $L_A(t)$ and $L_P(t)$ denote actual and predicted loads at sample time t, respectively. The general performance measures include MAE, MAPE, and SD. MASE provides a scale-free error metric for comparing forecasting methods on a single series [36]. In (6), the numerator $\mathrm{MAE}(k)_{\text{out-of-sample}}$ for k-step out ($k = 1, \ldots, 12$) is calculated for the multistep WNN forecasts computed out-of-sample (in the testing data set). The denominator $\mathrm{MAE}_{\text{in-sample}}$ is calculated for the one-step "naïve forecast" computed in-sample (in the training and validation data sets). The naïve forecast for each future period is the actual value for the previous period [37]. This denominator $\mathrm{MAE}_{\text{in-sample}}$ is used to scale the numerator $\mathrm{MAE}(k)_{\text{out-of-sample}}$ to generate a scale-free error metric that is stable, easy to compute, and in the correct unit. If the MASE value is less than one, this indicates that the forecast of the presented method is better than the one-step naive forecast. However, if the MASE value is greater than one, this indicates the opposite. Multistep MASE values are often larger than one as the forecasting horizon increases because one step naive forecast is used for scaling [36], [37]. Equations (5)–(8) can also be applied to moving forecasts with multiple WNNs.

Two examples are presented to demonstrate our method. Example 1 uses a classroom-type problem to compare a single (standard) NN to our two-level wavelet NNs so that our method can be duplicated and verified in a simple way. Example 2 demonstrates the values of spike filtering methods, two-level decomposition, Db4 wavelet, Symmetrization padding, selected time and date indices (hourly, weekly, monthly, and sunset time), and relative increment transformation to the LL frequency component.

In both examples, standard neural networks based on the back-propagation learning algorithm in [34, Ch. 4] are used. The training, validation, and test processes in a three-way data split are used to determine and demonstrate the parameters in the model. All NNs are trained offline by using historical data with weights randomly initialized, and the training terminates when the stopping criteria is reached to be described in Examples 1 and 2. These NNs are then trained online with the latest 12 loads as explained in Section IV-C.

Example 1: Consider the signal:

$$y(t) = 100 \sin(2\pi 10t) + 20 \sin(2\pi 150t) + \sin(2\pi 200t) \tag{9}$$

where the signal $y(t)$ is composed of a low frequency component $100\sin(2\pi 10t)$, a medium component $20\sin(2\pi 150t)$, and a high component $\sin(2\pi 200t)$. The signal is similar to the actual load in terms of relative amplitude and frequency. A total of 3600 data points $(t, \tilde{y}(t))$ were randomly generated:

$$\tilde{y}(t) = y(t) + \varepsilon(t) \tag{10}$$

where $t \in [1, \ldots, 3600]$ and $\{\varepsilon(t)\}$ were independent and identically distributed normal noises with zero mean and unit variance $N(0,1)$. The first one-third of data points were used for training, the second one-third of data for validation, and the last one-third of data for testing.

A single NN without wavelet decomposition is compared to neural networks with two-level wavelet decomposition. The relative increment transformation is not used for this example because $y(t)$ consists of three sine functions which are periodical, and there is no need to use this transformation to make $\{y(t)\}$ stationary. Based on the training, validation, and test processes in a three-way data split, the number of hidden neurons for the standard NN method is set to be 11, and the numbers of hidden neurons for our method are set to be 8, 7, and 13 for H, LH, and LL NNs, respectively. For both methods, NN training processes stop when MAE thresholds (stopping criteria) are reached. From the test data set, the overall MAE and SD are, respectively, 1.73 and 2.33 for standard NN method, whereas the overall MAE and SD are, respectively, 0.85 and 1.06 for our method. MAPE is not adopted since $\{y(t)\}$ may have zero values. MAEs and SDs indicate that the predictions obtained from using two-level wavelet NNs are both closer to the true values in data series $\{y(t)\}$ and have smaller standard deviations than the ones obtained using a single NN.

Example 2: Wavelet neural networks with spike filtering are tested with system load data provided by ISO New England. The training period is from January 1, 2007 to December 31, 2007, the validation period is from January 1, 2008 to June 30, 2008, and the test period is from July 1, 2008 to December 31,

TABLE I
MAEs (MW) FOR MULTIPLE WNNs

Min.	WNN_1	WNN_2	WNN_4	WNN_6	WNN_8	WNN_{10}	WNN_{12}
5	13.43	13.50	13.49	13.45	13.46	13.42	13.48
10	19.90	19.97	19.95	19.97	19.94	19.90	19.99
15	25.60	25.75	25.75	25.73	25.71	25.59	25.61
20	31.56	31.68	31.79	31.73	31.72	31.55	31.57
25	36.88	36.99	37.07	36.99	37.07	36.87	36.81
30	42.48	42.65	42.75	42.66	42.70	42.46	42.40
35	48.09	48.29	48.43	48.31	48.31	48.06	47.99
40	53.83	54.09	54.18	54.05	54.05	53.79	53.66
45	59.23	59.55	59.59	59.49	59.56	59.22	59.01
50	64.74	65.18	65.21	65.16	65.21	64.73	64.55
55	69.26	69.71	69.63	69.69	69.69	69.24	69.11
60	74.40	74.86	74.78	74.97	74.89	74.38	74.27

2009. Ten cases are tested. Since there are many factors in setting the forecasting model, and each factor has multiple options, the number of possible combinations of options is very large. To have a practical way to demonstrate the appropriateness of options selected for individual factors, the configuration determined through training, validation, and test processes is treated as the nominal configuration. Based on it, each factor is then examined in individual cases below. Cases 1–7 are for training and validation: Case 1 for micro and macro spike filtering, Case 2 for spike filtering thresholds, Case 3 for decomposition levels, Case 4 for selecting Daubechies wavelets, Case 5 for padding strategies, Case 6 for date and time indices, and Case 7 for relative increment transformation. Cases 8–10 are for testing: Case 8 for test results and prediction interval construction, Case 9 for comparing with ISO-NE's method, and Case 10 for Monte Carlo simulations.

To reduce computation time, Cases 3–7 are based on WNN_1 because its results are very similar to the individual results from other WNNs as reported in Table I, while the other cases are based on the 12 dedicated WNNs. For all the cases, there are three layers in all the neural networks: one input layer, one hidden layer, and one output layer. Through training, validation, and test processes in a three-way data split, the numbers of hidden neurons are 6, 13, and 18 for H, LH, and LL NNs, respectively. They are not identical because the decomposed load components have different features. Based on testing, a single WNN is trained offline for 3 h (stopping criterion), and 12 WNNs require a total of 36 h for training offline.

Case 1: Spike filtering methods are tested with ISO-NE's real-time load data. Results for multiple WNNs with the loads filtered by the micro and macro filters are compared to the ones with unfiltered loads, the loads only filtered by the micro filter in 4 s, and the loads only filtered by the macro filter. The results for 5- to 60-min outs in Table II show that both micro filtering in 4 s and macro filtering improve MAEs and SDs. Furthermore, using the micro and macro spike filtering together produces the smallest MAEs and SDs, and these results are treated as nominal ones and will be used later in Cases 8 and 10 for comparisons.

Case 2: To detect spikes by micro or macro filtering, three thresholds m, w_1, and w_2 should be determined. Based on observation, spike magnitudes are usually greater than 40 MW for ISO-NE's load data, the widths of micro spikes are less than 3 points, and the widths of macro spikes are less than 10 points. Through testing based on a three-way data split, the nominal values for m, w_1, and w_2 are set to be 50, 3, and 10, respectively. To partially validate this choice, different values of m are examined when w_1 and w_2 are fixed at their nominal values. MAEs and SDs in Table III show that the results with different m values are quite similar, and the configuration with $m = 50$ produces the best forecasting accuracy. The same steps are separately taken for the widths w_1 and w_2, and 3 and 10 are chosen, respectively.

Case 3: Wavelet decomposition results from zero level (a single NN without wavelet decomposition) to three levels are

TABLE II
MAPEs (%), MAEs (MW), AND SDs (MW) FOR MULTIPLE WNNs IN MOVING FORECASTS WITH AND WITHOUT SPIKE FILTERING METHODS

Min.	With loads not filtered			With loads only filtered by the micro filter in four seconds		
	MAPE	MAE	SD	MAPE	MAE	SD
5	0.10	15.56	17.77	0.09	13.56	16.24
10	0.15	22.78	27.09	0.13	20.08	25.04
15	0.23	35.56	41.94	0.17	25.95	32.47
20	0.31	47.69	56.36	0.21	32.00	39.57
25	0.38	57.06	68.12	0.25	37.46	46.47
30	0.44	67.32	80.83	0.29	43.27	53.54
35	0.53	80.05	95.49	0.32	49.09	60.40
40	0.61	92.63	110.24	0.36	55.01	67.24
45	0.68	103.96	123.73	0.40	66.60	73.90
50	0.76	115.77	137.60	0.44	66.47	80.65
55	0.79	121.35	142.87	0.47	71.16	85.83
60	0.85	129.30	151.05	0.50	76.54	91.73
Min.	With loads only filtered by the macro filter			With loads filtered by the micro and macro filters		
	MAPE	MAE	SD	MAPE	MAE	SD
5	0.09	13.52	16.19	0.09	13.49	16.03
10	0.13	20.05	24.99	0.13	20.00	24.43
15	0.17	25.92	32.41	0.17	25.65	30.71
20	0.21	31.97	39.51	0.21	31.61	36.89
25	0.25	37.42	46.40	0.24	36.88	42.97
30	0.29	43.23	53.46	0.28	42.45	49.13
35	0.32	49.04	60.34	0.32	48.05	55.20
40	0.36	54.97	67.20	0.36	53.71	61.25
45	0.40	60.56	73.80	0.39	59.06	66.87
50	0.44	66.41	80.51	0.42	64.59	72.46
55	0.47	71.09	85.67	0.45	69.14	77.85
60	0.50	76.47	91.56	0.49	74.28	83.57

TABLE III
MAEs (MW) AND SDs (MW) FOR SPIKE FILTERING METHODS WITH DIFFERENT m VALUES

Min.	m=45		m=50		m=55		m=60	
	MAE	SD	MAE	SD	MAE	SD	MAE	SD
5	13.6	16.3	13.5	16.0	13.5	16.2	13.5	16.3
10	20.0	25.0	20.0	24.4	20.0	24.9	20.0	25.1
15	25.9	32.1	25.7	30.7	25.9	32.4	25.9	32.4
20	31.8	39.1	31.6	36.9	31.9	39.7	32.0	39.5
25	37.2	45.5	36.9	43.0	37.3	46.3	37.3	46.0
30	42.8	52.1	42.5	49.1	43.0	53.2	43.0	52.7
35	48.5	58.9	48.1	55.2	48.8	60.7	48.7	59.8
40	54.3	65.5	53.7	61.3	54.7	67.9	54.6	66.6
45	59.8	71.7	59.1	66.9	60.3	74.7	60.1	73.1
50	65.5	78.1	64.6	72.5	66.0	81.5	65.8	79.6
55	70.0	83.3	69.1	77.9	70.5	86.6	70.3	84.8
60	75.2	89.1	74.3	83.6	75.9	92.3	75.5	90.6

TABLE IV
MAEs (MW) AND SDs (MW) FOR WNN_1 WITH DIFFERENT DECOMPOSITION LEVELS

Min.	0-Level	1-Level		2-Levels		3-Levels	
	MAE	MAE	SD	MAE	SD	MAE	SD
5	15.64	17.94	31.80	13.43	15.92	19.66	55.02
10	24.83	25.54	35.46	19.90	24.40	27.16	57.66
15	32.89	30.93	38.56	25.60	30.85	30.57	58.63
20	40.24	36.47	42.44	31.56	37.19	36.46	61.16
25	47.64	41.91	46.65	36.88	43.29	41.90	63.62
30	54.93	47.18	51.42	42.48	49.47	47.55	67.42
35	62.96	53.44	56.40	48.09	55.42	52.48	71.19
40	70.78	59.58	61.62	53.83	61.38	58.14	75.00
45	78.52	65.25	67.63	59.23	67.08	63.42	79.94
50	86.41	71.26	73.26	64.74	72.75	69.38	85.17
55	94.06	78.20	79.49	69.26	78.12	74.06	89.05
60	102.04	84.38	85.44	74.40	83.85	79.11	93.59

TABLE V
MAEs (MW) FOR WNN_1 WITH DIFFERENT DAUBECHIES WAVELETS

Min.	Db2	Db4	Db6	Db8	Db12	Db20
5	27.43	13.43	16.83	17.93	17.15	17.65
10	29.88	19.90	25.51	25.66	25.57	26.25
15	35.71	25.60	32.40	33.43	33.28	34.33
20	39.59	31.56	39.09	39.70	39.97	41.05
25	52.42	36.88	45.20	46.33	46.59	47.62
30	56.03	42.48	51.40	52.89	52.90	53.63
35	61.08	48.09	57.74	59.62	59.52	60.41
40	66.42	53.83	64.80	67.40	66.74	67.90
45	80.91	59.23	72.19	74.89	73.87	75.40
50	85.71	64.74	79.68	83.02	81.34	83.58
55	89.69	69.26	87.29	89.08	88.31	91.25
60	95.39	74.40	94.93	93.87	94.27	97.23

compared. MAEs presented in Table IV show that two-level wavelet neural networks produce the best forecasting accuracy.

SDs for decomposition levels one to three show insignificant differences, and hence will not be given in Cases 4–7.

Case 4: Based on the two-level wavelet decomposition, results using different Daubechies wavelets (Db2-Db20) are compared and are partially reported in Table V. MAEs indicate that the Db4 gives the best prediction accuracy. This is consistent with the analysis in Section IV-B.

Case 5: To handle distortions, different padding strategies are used, including zero padding, periodic padding with order one, and symmetrization padding. In Table VI, results using different padding strategies are compared, and MAEs show that the symmetrization strategy gives the best prediction accuracy.

Case 6: Beyond load inputs to NNs, the selections of time and date indices are investigated and reported in Table VII. The combination which includes the loads of the last hour (LD), the hourly index (HI), the weekly index (WI), the monthly index (MI), and the sunset time index (SI) gives the smallest MAEs when compared to other combinations.

Case 7: Components with and without relative increment transformation applied are tested. The strategy using an LL frequency component with RI, and LH and H frequency components without RI produces the smallest MAEs (e.g., 75 MW for the 60-min out) when compared to the other strategies of using LL without RI, and LH and H with RI (e.g., 300 MW for the 60-min out), and using LL, LH, and H with and without RI (very large).

TABLE VI
MAEs (MW) FOR THE WNN_1 WITH DIFFERENT PADDING STRATEGIES

Min.	Zero	Periodic	Symmetrization
5	7250.92	668.44	13.43
10	7251.40	670.97	19.90
15	7251.77	675.08	25.60
20	7252.18	678.99	31.56
25	7252.63	682.63	36.88
30	7253.07	686.52	42.48
35	7470.42	689.50	48.09
40	7263.67	692.31	53.83
45	7132.73	690.89	59.23
50	6981.50	675.37	64.74
55	6308.64	789.13	69.26
60	7053.08	1481.67	74.40

TABLE VII
MAEs (MW) FOR WNN_1 WITH DIFFERENT TIME INDICES

Min.	LD	LD+HI	LD+HI +WI	LD+HI +WI+MI	LD+HI+WI +MI+SI
5	35.18	29.57	21.39	19.34	13.43
10	56.64	46.10	32.46	29.08	19.90
15	79.20	59.14	42.94	38.03	25.60
20	103.78	72.01	52.56	46.06	31.56
25	127.42	86.01	63.02	53.93	36.88
30	151.52	100.10	73.84	62.08	42.48
35	183.59	116.29	84.84	70.40	48.09
40	220.04	136.21	100.10	83.61	53.83
45	247.66	150.34	107.99	88.09	59.23
50	285.25	172.55	125.81	102.34	64.74
55	301.70	177.87	129.31	103.09	69.26
60	328.09	191.38	140.44	111.18	74.40

TABLE VIII
MASEs, MAPEs (%), MAEs (MW), AND SDs (MW) FOR OUR METHOD

Min.	MAPE	MAE	SD	MASE
5	0.09	12.52	14.61	0.23
10	0.13	18.45	19.87	0.35
15	0.16	23.72	24.67	0.45
20	0.20	29.36	30.15	0.56
25	0.24	34.49	35.48	0.65
30	0.27	39.89	41.15	0.76
35	0.31	45.36	46.48	0.86
40	0.33	51.06	52.13	0.97
45	0.35	56.32	57.52	1.07
50	0.38	61.80	63.23	1.17
55	0.42	66.06	67.77	1.25
60	0.45	71.02	72.93	1.35

Case 8: Cases 1–7 are for training and validation, and Cases 8–10 are for testing. As shown in Table VIII, the small MAPEs, MAEs, and SDs are close to the nominal results (in the block with the loads filtered by the micro and macro filters) in Table II, indicating that the parameters are properly selected. Also, the MASEs for 5- to 40-min outs are less than one, indicating that our multistep forecasts are better than the one-step naive forecast. The MASEs for 45- to 60-min outs are slightly greater than one. This corresponds to the explanation in the beginning of Section V that multistep MASE values will often be larger than one as the forecasting horizon increases.

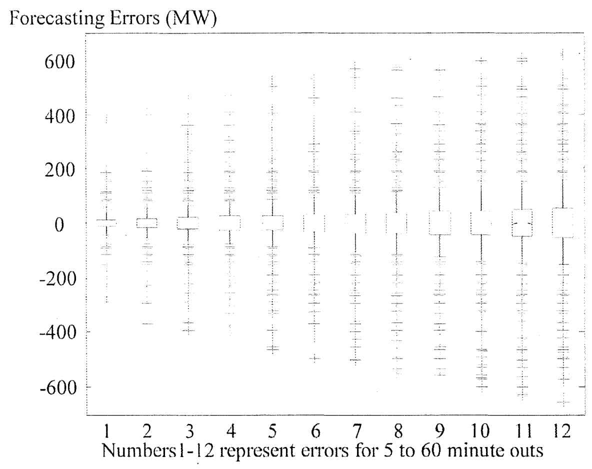


Fig. 8. Box plots for forecasting errors for 5- to 60-min outs.

To evaluate the errors for 5- to 60-min outs, box plots, as depicted in Fig. 8, are used to graphically depict errors through five-number summaries: sample minimum, lower quartile, median, upper quartile, and sample maximum [38]. The figure shows that forecasting errors for 5- to 60-min outs by our method have near zero medians, and box shapes are almost symmetric. The range, especially the outlier and inter-quartile ranges, however, gradually expand as the minute out increases, due to the fact that data uncertainty increases from 5 to 60 min in 5-min steps.

To evaluate the error bias, average errors (the mean of the differences between the actual and predicted loads) for 5- to 60-min outs are calculated to be from -2.1 MW to 0 MW. This range is relatively insignificant compared to the overall load range, from 9000 MW to 27 000 MW. This indicates that the model is almost unbiased. Furthermore, the percentages of under and over forecasts are nearly 50% for both.

To empirically construct prediction intervals, consider 5-min outs as an example. At time t, historical 5-min errors (actual minus predicted loads) before time t are ordered. For a nominal coverage rate 1-α, e.g., $\alpha = 0.1$, the lower and upper bounds of the 90% prediction interval are determined and then added to the forecast at time t to be the approximated prediction interval. For our testing, the errors from July 1, 2008 to November 30, 2008 (>40 000 errors) and the prediction are used to construct the prediction interval for $t =$ 00:05 am on December 1, 2008. The lower and upper bounds obtained are -112.75 MW and 104.52 MW, respectively, and the predicted load is 10 619 MW. When the error at 00:05 am is available, this new error and previous errors are then used together for $t =$ 00:10 am. To quantify forecasting accuracy, this process repeats until the end of December. It turns out that 87.02% of actual load data falls within approximated prediction intervals (i.e., actual percentage coverage = 87.02%). This is close to 90%, indicating that approximated prediction intervals are reasonably accurate. The same steps are taken for 10- to 60-min outs, and similar results are obtained.

Prediction intervals can also be obtained based on an estimated distribution of the variable to be forecasted by using, for example, a modified bootstrap method as presented in [39] for short-term load forecasting, or an adapted resampling method as presented in [40] for wind power generation forecasting. Since the method we used provided reasonably accurate results, these methods in [39] and [40] are not explored.

TABLE IX
MAPES (%) AND MAES (MW) COMPARING OUR METHOD' S RESULTS TO ISO-NE'S RESULTS

Min.	ISO-NE's Method		Our Method	
	MAPE	MAE	MAPE	MAE
5	0.26	43.74	0.08	14.37
10	0.30	50.68	0.13	21.57
15	0.34	57.99	0.16	27.80
20	0.38	64.58	0.20	34.17
25	0.43	72.29	0.23	40.06
30	0.48	80.95	0.27	46.44
35	0.53	90.43	0.31	52.74
40	0.60	100.76	0.35	59.21
45	0.64	109.41	0.38	65.46
50	0.70	119.12	0.42	71.83
55	0.75	127.81	0.45	77.43
60	0.81	138.33	0.49	83.54

Case 9: Results of our method and of ISO-NE's method in [3] reviewed in Section II are compared based on ISO-NE's real-time data. The forecasting period for comparison is from July 1, 2008 to July 31, 2008. MAPEs and MAEs in Table IX show that our method produces smaller errors than ISO-NE's. This demonstrates that our method is significantly better than ISO-NE's.

Beyond the comparison above, it is difficult to compare our results to others since there is no standard test data set for a fair comparison. Nevertheless, the following results have been reported in the literature: the MAPEs for a United States power utility [4] range from 0.4% to 1.1% for 20- 60-min outs in 10-min periods; the MAPEs for British electricity demand [20] range from 0.1%–0.5% for 1- to 30-min outs in 1-min periods; the average MAPEs for 12 5-min periods in the Ubatuba area in Brazil [18] are 2.62%, 0.39%, and 18.72% for ARIMA, NN, and the adaptive neuro-fuzzy system, respectively; the MAPEs for a 5-min out for the state of New South Wales in Australia [25] are 0.27%, 0.28%, 0.33%, and 0.27% for least regression, least mean square, BPNN, and support vector regression, respectively. Since data features as well as forecasting resolutions and periods are different from paper to paper, and implementation details are not open, it is difficult to evaluate individual performances. However, our method seems to be very competitive.

Case 10: To test the robustness of our method, two sets of Monte Carlo simulations are performed each with $N = 20$ simulations. The first set of Monte Carlo simulations is run with a random weight initialization. Since N simulations are independent, the mean μ and standard deviation σ are calculated for MAPEs, MAEs, and SDs. Results in Table X show that the means μ_{MAPE}, μ_{MAE}, and μ_{SD} for 5- to 60-min outs are close to the nominal MAPE, MAE, and SD in the loads filtered by the micro and macro filters for all the cells, as reported in Table II. Also, the standard deviations σ_{MAPE}, σ_{MAE}, and σ_{SD} are small. This indicates that our method is robust.

The second set of Monte Carlo simulations is run with a random re-sampling step [35]. For example, time t is randomly selected from the test data set, and then historical data from

TABLE X
MEANS AND STANDARD DEVIATIONS FOR MAPEs (%), MAEs (MW), AND SDs (MW) FROM MONTE CARLO SIMULATIONS WITH A RANDOM WEIGHT INITIALIZATION (WITH $N = 20$ SIMULATIONS)

Min.	μ_{MAPE}	σ_{MAPE}	μ_{MAE}	σ_{MAE}	μ_{SD}	σ_{SD}
5	0.09	0.00	12.81	0.64	16.42	3.42
10	0.13	0.00	19.29	2.27	21.59	2.76
15	0.16	0.00	23.93	0.45	25.94	2.30
20	0.20	0.00	29.56	0.47	31.19	1.99
25	0.24	0.01	34.70	0.59	36.42	1.83
30	0.27	0.01	40.17	0.74	41.97	1.80
35	0.31	0.01	45.65	0.88	47.32	1.98
40	0.35	0.01	51.36	1.06	53.01	2.29
45	0.38	0.01	56.68	1.24	58.42	2.65
50	0.42	0.01	62.21	1.47	64.14	3.09
55	0.45	0.01	66.51	1.54	68.67	3.16
60	0.48	0.01	71.51	1.67	73.82	3.36

TABLE XI
MEANS AND STANDARD DEVIATIONS FOR MAPEs (%), MAEs (MW), AND SDs (MW) FROM MONTE CARLO SIMULATIONS WITH RANDOM RE-SAMPLING STEPS (WITH $N = 20$ SIMULATIONS)

Min.	μ_{MAPE}	σ_{MAPE}	μ_{MAE}	σ_{MAE}	μ_{SD}	σ_{SD}
5	0.09	0.00	12.43	0.81	13.58	4.38
10	0.13	0.01	18.43	1.34	19.28	3.64
15	0.17	0.01	23.75	2.01	24.60	3.32
20	0.21	0.01	29.52	2.82	30.49	3.63
25	0.24	0.02	34.76	3.54	35.94	4.10
30	0.28	0.02	40.35	4.31	41.80	4.83
35	0.32	0.03	46.20	5.44	47.45	5.92
40	0.36	0.03	52.22	6.55	53.83	7.13
45	0.40	0.04	57.74	7.72	59.26	8.35
50	0.44	0.05	63.40	8.74	65.34	9.70
55	0.47	0.05	67.74	9.29	69.72	10.18
60	0.50	0.05	72.90	10.04	75.15	10.89

one-year before t are used for training offline, and the loads one month after t are to be predicted. In comparison to the results using a random weight initialization, results in Table XI show that the means μ_{MAPE}, μ_{MAE}, and μ_{SD} for 5- to 60-min outs are close to the ones in Table X. The standard deviations σ_{MAPE}, σ_{MAE}, and σ_{SD} are slightly larger than the ones in Table X for most of the cells, due to the complicated load features. However, the standard deviations for the random re-sampling step are still small. This indicates that our method is robust, and data sets are not sparse.

VI. CONCLUSION

This paper presents a method of wavelet neural networks with data pre-filtering to forecast very short-term loads 1 h into the future in 5-min steps in a moving window manner. The spike filtering methods remove spikes in real-time. This WNN method can capture the load components at different frequencies. Daubechies-4 with two-level decomposition is the best configuration, which balances the decomposed level, the filter length, and the minimum padding length for decomposition. Symmetrization is shown to be the best strategy to handle the distortion. Applying the relative increment transformation to load series enhances the load stationarity. Based on test results, 12 dedicated wavelet neural networks are used to perform moving forecasts every 5 min. Numerical testing shows accurate predictions with small standard deviations for VSTLF based on the data set from ISO New England.

ACKNOWLEDGMENT

The authors would like to thank Dr. Y. Chen at the Edison Mission Marketing and Trading, Dr. Y. Ou at the ISO New England, Dr. K. W. Cheung at the Alstom Grid, Dr. N. Ravishanker at the University of Connecticut, Dr. R. J. Hyndman at the Monash University, and Dr. Z. Chi at the University of Connecticut. The authors would also like to thank anonymous reviewers whose comments contributed to improve the quality of this paper.

REFERENCES

[1] C. Guan, P. B. Luh, M. A. Coolbeth, Y. Zhao, L. D. Michel, Y. Chen, C. J. Manville, P. B. Friedland, and S. J. Rourke, "Very short-term load forecasting: Multilevel wavelet neural networks with data pre-filtering," in *Proc. IEEE Power and Energy Society 2009 General Meeting*, Calgary, AB, Canada, Jul. 2009.

[2] K. Liu, S. Subbarayan, R. R. Shoults, M. T. Manry, C. Kwan, F. L. Lewis, and J. Naccarino, "Comparison of very short-term load forecasting techniques," *IEEE Trans. Power Syst.*, vol. 11, no. 2, pp. 877–882, May 1996.

[3] P. Shamsollahi, K. W. Cheung, Q. Chen, and E. H. Germain, "A neural network based very short-term load forecaster for the interim ISO New England electricity market system," in *Proc. Innovative Computing for Power—Electric Energy Meets the Market: 22nd IEEE Power Engineering Society Int. Conf. Power Industry Computer Applications*, Sydney, Australia, May 2001.

[4] W. Charytoniuk and M. S. Chen, "Very short-term load forecasting using artificial neural networks," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 263–268, Feb. 2000.

[5] K. Xie, F. Wang, and E. Yu, "Very short-term load forecasting by Kalman filter algorithm," *Proc. Chinese Soc. Elect. Eng.*, vol. 16, no. 4, pp. 245–249, Jul. 1996.

[6] Y. Chen, P. B. Luh, C. Guan, Y. G. Zhao, L. D. Michel, M. A. Coolbeth, P. B. Friedland, and S. J. Rourke, "Short-term load forecasting: Similar day-based wavelet neural networks," *IEEE Trans. Power Syst.*, vol. 25, no. 1, pp. 322–330, Feb. 2010.

[7] Y. Zhao, P. B. Luh, C. Bomgardner, and G. H. Beerel, "Short-term load forecasting: Multilevel wavelet neural networks with holiday corrections," in *Proc. IEEE Power and Energy Society 2009 General Meeting*, Calgary, AB, Canada, Jul. 2009.

[8] B. D. Ripley, *Neural Networks and Pattern Recognition*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

[9] J. N. Fidalgo and J. A. P. Lopes, "Load forecasting performance enhancement when facing anomalous events," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 408–415, Feb. 2005.

[10] W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: Properties and applications in non-Gaussian signal processing," *IEEE Trans. Signal Process.*, vol. 55, no. 11, pp. 5286–5298, Nov. 2007.

[11] R. J. Bessa, V. Miranda, and J. Gama, "Entropy and correntropy against minimum square error in offline and online three-day ahead wind power forecasting," *IEEE Trans. Power Syst.*, vol. 24, no. 4, pp. 1657–1666, Nov. 2009.

[12] B. Fox, D. Flynn, L. Bryans, N. Jenkins, D. Milborrow, M. O'Malley, R. Watson, and O. Anaya-Lara, "Wind power integration connection and system operational aspects," *IET Power Energy Ser.*, 2007.

[13] F. Wang, K. Xie, E. Yu, G. Liu, and M. Wang, "A simple and effective ultra-short term load forecasting method," *Power Syst. Technol.*, vol. 20, no. 3, pp. 41–43, Mar. 1996.

[14] D. Luo and H. He, "A shape similarity criterion based curve fitting algorithm and its application in ultra-short-term load forecasting," *Power Syst. Technol.*, vol. 31, no. 21, pp. 81–84, Nov. 2007.

[15] J. Zhou, B. Zhang, J. Shang, J. Yao, and M. Cheng, "Very short-term load forecast based on multi-sample extrapolation and error analysis," *Elect. Power Autom. Equip.*, vol. 25, no. 2, pp. 15–21, Feb. 2005.

[16] Z. Yang, G. Tang, Y. Song, and R. Cao, "Improved cluster analysis based ultra-short term load forecasting method," *Autom. Elect. Power Syst.*, vol. 29, no. 24, pp. 83–86, Dec. 2005.

[17] J. Lu, X. Zhang, and W. Sun, "A real-time adaptive forecasting algorithm for electric power load," in *Proc. 2005 IEEE PES Transmission and Distribution Conf. Expo.: Asia and Pacific*, Dalian, China, 2005.

[18] L. C. M. de Andrade and I. N. da Silva, "Using intelligent system approach for very short-term load forecasting purposes," in *Proc. 2010 IEEE Int. Energy Conf. Exhib.*, Manama, Bahrain, Dec. 2010.

[19] A. Setiawan, I. Koprinska, and V. G. Agelidis, "Very short-term electricity load demand forecasting using support vector regression," in *Proc. 2009 Int. Joint Conf. Neural Networks*, Atlanta, GA, Jun. 2009.

[20] J. W. Taylor, "An evaluation of methods for very short-term load forecasting using minute-by-minute British data," *Int. J. Forecast.*, vol. 24, pp. 645–658, 2008.

[21] D. J. Trudnowski and W. L. Mcreynolds, "Real-time very short-term load prediction for power system automatic generation control," *IEEE Trans. Control Syst. Technol.*, vol. 9, no. 2, pp. 254–260, Mar. 2001.

[22] H. Yang, H. Ye, G. Wang, J. Khan, and T. Hu, "Fuzzy neural very short-term load forecasting based on chaotic dynamics reconstruction," *Chaos, Solitons & Fractals*, vol. 29, pp. 462–469, 2006.

[23] S. Kawauchi, H. Sugihara, and H. Sasaki, "Development of very short-term load forecasting based on chaos theory," *Elect. Eng. Jpn.*, vol. 148, no. 2, pp. 55–63, 2004.

[24] L. C. M. de Andrade and I. N. da Silva, "Very short-term load forecasting using a hybrid neuro-fuzzy approach," in *Proc. 2010 11th Brazilian Symp. Neural Networks*, Sao Carlos, Brazil, Oct. 2010.

[25] I. Koprinska, R. Sood, and V. Agelidis, "Variable selection for five-minute ahead electricity load forecasting," in *Proc. 20th Int. Conf. Pattern Recognition*, Istanbul, Turkey, Aug. 2010.

[26] S. W. Smith, *Scientist and Engineer's Guide to Digital Signal Processing*, second ed. san Diego, CA: California Technical Publishing, 1999.

[27] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach*, 3rd ed. New York: McGraw-Hill, 2006.

[28] A. J. R. Reis and A. P. A. da Silva, "Feature extraction via multi-resolution analysis for short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 189–198, Feb. 2005.

[29] D. Benaouda, F. Murtagh, J. L. Starck, and O. Renaud, "Wavelet-based nonlinear multi-scale decomposition model for electricity load forecasting," *Neurocomputing*, vol. 70, pp. 139–154, 2006.

[30] S. G. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. New York: Academic, 2009.

[31] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, 2nd ed. Wellesley, MA: Wellesley-Cambridge Press, 1997.

[32] C. Guan and P. B. Luh, Derivation of the Padding Length for Daubechies Wavelets, 2010, working paper, referred upon request.

[33] Weather Forecasting, The Weather World 2010 Project. [Online]. Available: ww2010.atmos.uiuc.edu/%28Gl%29/guides/mtr/fcst/mth/oth.rxml.

[34] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Englewood Cliffs, NJ: Prentice Hall, 2009.

[35] M. Herrera, L. Torgo, J. Izquierdo, and R. Pérez-García, "Predictive models for forecasting hourly urban water demand," *J. Hydrol.*, vol. 387, pp. 141–150, 2010.

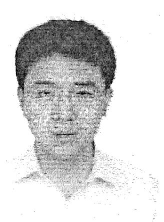
[36] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecast.*, vol. 22, pp. 679–688, 2006.

[37] R. J. Hyndman, "Another look at forecast accuracy metrics for intermittent demand," *Foresight: Int. J. Appl. Forecast.*, no. 4, pp. 43–46, Jun. 2006.

[38] AP Statistics: Boxplots (Aka, Box and Whisker Plots), Stat Trek. [Online]. Available: stattrek.com/ap-statistics-1/boxplot.aspx.

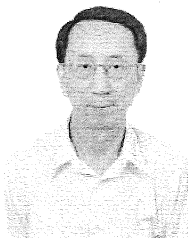
[39] S. Fan and R. J. Hyndman, "Short-term load forecasting based on a semi-parametric additive model," *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 134–141, Feb. 2012.

[40] P. Pinson and G. Kariniotakis, "Conditional prediction intervals of wind power generation," *IEEE Trans. Power Syst.*, vol. 25, no. 4, pp. 1845–1856, Nov. 2010.



Che Guan (S'09) received the B.S. degree from Changchun University of Science and Technology, Changchun, China, in 2004, and the M.S. degree from Chinese Academy of Sciences, Beijing, China, in 2007. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Connecticut, Storrs.

His research interests include power systems, control systems and optimization, intelligent systems, statistical algorithms, and signal processing.



Peter B. Luh (F'95) received the B.S. degree from National Taiwan University, Taipei, the M.S. degree from the Massachusetts Institute of Technology, Cambridge, and the Ph.D. degree from Harvard University, Cambridge.

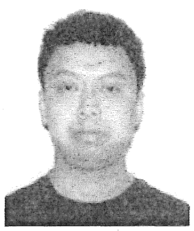
He has been with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, since 1980, and currently is the SNET Professor of Communications & Information Technologies. He served as the Head of the Department from 2006 to 2009. He is also a member of the Chair Professors Group in the Department of Automation, Tsinghua University, Beijing. His interests include Power Systems—design of auction methods for electricity markets, robust wind integration to the grid, and electricity load and price forecasting with demand management; Manufacturing Systems—planning, scheduling, and coordination of design, manufacturing, and service activities; Smart Buildings—optimized energy management, HVAC fault detection and diagnosis, and emergency crowd guidance for green and safe buildings; and mathematical optimization of large-scale mixed-integer problems, and decision-making under uncertain, distributed, or antagonistic environments.

Prof. Luh is a member of the IEEE Periodicals Committee, and the Senior Advisor on Automation for the Robotics and Automation Society. He was VP Publication Activities for the IEEE Robotics and Automation Society, the Editor-in-Chief of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, and the founding Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.

Laurent D. Michel received the M.S. and Ph.D. degrees in computer science from Brown University, Providence, RI, in 1996 and 1999, respectively.

He is an Associate Professor of computer science and engineering at the University of Connecticut, Storrs. He is also a Principal Analyst at Voting Systems Security, LLC. His interests span combinatorial optimization with a particular emphasis on constraint programming, load forecasting, and voting technology. He has co-authored two monographs and more than 80 papers.

Prof. Michel sits on the Editorial Board of *Constraints*, *Mathematical Programming Computation*, and *Constraint Letters*.



Yuting Wang received the B.S. and M.S. degrees in electric engineering from Shanghai Jiaotong University, Shanghai, China, in 2006 and 2009, respectively, and the M.S. degree in the Computer Science & Engineering Department at the University of Connecticut, Storrs, in 2011. He is pursuing the Ph.D. degree in the Computer Science and Engineering Department at the University of Minnesota, Minneapolis.

His research interests include programming languages design, computational logic, proof theory, and formal verification of computational systems.

Peter B. Friedland (M'92) received the B.S.E.E. degree from the University of Connecticut, Storrs, in 1986.

He has been working in the electric utility industry for the majority of his career including work with an EMS Vendor, Transmission Operator, and ISO/RTO. He was employed by ISO New England in varying management capacities including Standard Market Design (SMD) Project Manager, IT-EMS Manager, and Operations Manager. He was employed by GE Digital Energy as a Smart Grid Director. He has recently been consulting for the ISO/RTO industry and other start-up companies in the connected energy space.

Mr. Friedland is a past member of the University of Connecticut ECE Industrial Advisory Board.