Event Detection in NILM using Cepstrum smoothing

Leen De Baets, Joeri Ruyssinck, Dirk Deschrijver, Tom Dhaene
Ghent University - iMinds
Email: leen.debaets@intec.ugent.be

***Abstract*—Event detection plays an important role in non-intrusive load monitoring to accurately detect the switching of appliances in a residential environment. Improving the detection ratios of those methods while keeping the computational cost under control is important. This paper presents a new event detection mechanism that works in the frequency domain and uses Cepstrum smoothing to eliminate noise. We explore the potential of our method by comparing with the χ^2 GOF method on the BLUED dataset. The results indicate that our method is competitive with the state-of-the-art having as advantage that the same feature can also be used for appliance detection.**

I. Introduction

Non-Intrusive Load Monitoring (NILM) concerns the analysis of the aggregate power consumption of electric loads in order to recognize the existence and the consumption profile of each individual appliance. NILM makes it possible to inform the consumers how much energy each appliance consumes thereby empowering them to reduce their energy consumption in an informed way.

In 1992, Hart was the first one to describe the steps of NILM [1]: 1) measuring the aggregated power consumption with a sensor attached to the main power cable, 2) detecting state-transitions of appliances (events) from the captured data, 3) clustering similar transitions using a well-chosen feature vector, 4) matching the on-transitions with the off-transitions, 5) recognizing and monitoring each appliance. These steps can be handled event-based or non-event-based [2]. The former starts with the state transition (or event) detection and the features around the detected transitions are used for clustering and classification. The latter does not rely on edge detection but uses every sample of the power trace for inferencing, which is typically more resource demanding.

In this paper an event-based method is presented that uses smoothed frequency components to detect an event. The remainder of this paper is structured as follows: in Section II a brief overview of related work is introduced, in Section III the proposed method is described and in Section IV, its performance is benchmarked and discussed.

II. State-of-the-art

Three efficient algorithms that are commonly used for real-time event detection in NILM are: the Generalized Likelihood Ratio (GLR) test [3], the chi-squared goodness-of-fit (χ^2 GOF) [4] and the CUmulative SUM (CUSUM) filtering [5]. To decide whether a certain event is present or not in a defined timeframe, GLR calculates a decision statistic from the natural log of a ratio of probability density functions before and after a potential change in mean. The χ^2 GOF test detects events by assuming that two consecutive timeframes share a common distribution. A χ^2 statistic is applied and an event is assumed if the null hypothesis is rejected. Finally, CUSUM is a method to determine changes in the quality number (e.g. the mean or the difference between the predicted and real value) by testing it against a criterion (a stop rule) describing when an event occurs. All three methods are statistical tests, work in the time domain and divide the signal into windows. In itself, these methods are not able to detect slow changes in the signal. However, this is possible if an extra analysis is added [5]. This step will keep track of the beginning and ending of transient behavior respectively defined as the moment when the stop rule of the CUSUM occurs in steady state and the moment when the stop rule of the CUSUM occurs in the transient state.

In addition to these statistical methods, more computational costly machine learning algorithms such as kernel clustering [6], Hidden Markov Models [7], Support Vector Machines [8], and Bayesian methods [9] have been proposed to address event detection. These methods contain parameters and require a training step to tune these in order to minimize misdetection rates. This training can be done in a supervised way requiring enough labelled data such that the algorithm calculates optimal parameters to predict these labels (representing the events) as accurately as possible. Training can also be done in an unsupervised manner requiring a cost function such that the algorithm with the optimal parameters has the lowest cost [6].

Cepstrum analysis was first introduced in 1963 where it was originally used to analyse the echoes within seismic signals produced from earthquakes [10]. Since then, it has proven to be a potent technique in several domains. One application is passive sonar which involves listening to the environment without sending signals in order to detect objects [11]. One specific example is the detection of fish under water. One problem is that this radiated signal is corrupted with multipath effects and interference noise. Applying Cepstrum analysis on

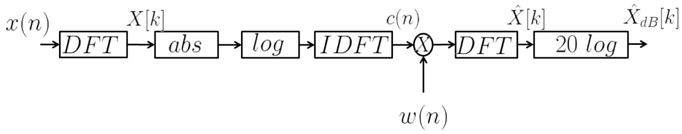


Fig. 1. A schematic overview of the transformation from a time signal to spectral smoothed dB-scaled frequency components.

the signal will alleviate this problem. Cepstrum analysis finds another application in speech recognition [12] where Cepstrum coefficients have been successfully applied to increase the robustness of various algorithms. Very recently it is also shown that Cepstrum coefficients can be used in a NILM setting as discriminative features in appliance recognition [13]. They are particularly interesting when multiple appliances are turned on or off at the same moment. In our work we explore and demonstrate the use of Cepstrum analysis for event detection.

III. Method

Events are detected in the frequency domain where smoothing occurs in the quefrency domain, rather than the time domain. The different steps are outlined in Figure 1. Consider a window x of length n from a power signal p,

$$x = \{p_i, p_{i+1}, ..., p_{i+n}\} \tag{1}$$

then the goal is to detect if there is an event or not. First, this window will be converted from the time to the frequency domain, by using the Fourier transform.

$$X[k] = \sum_{j=1}^{n} x[j] e^{-2\pi ikj/n} \ , \ 0 \le k < n \tag{2}$$

By transforming this information from the frequency domain into the quefrency domain, Cepstrum components can be computed. These can be computed by applying the inverse Fourier transform to the logarithm of $|X|$.

$$c[n] = \frac{1}{N} \sum_{k=0}^{N-1} log_{10}(\ |X[k]| \ ) e^{2\pi ink/N} \ , \ 0 \le n < N \tag{3}$$

These Cepstrum components are smoothed by means of a filter w, after which they are transformed back to frequency components by applying the Fourier transformation.

$$\hat{X}[k] = \sum_{j=1}^{n} w[j] \ c[j] e^{-2\pi ikj/n} \ , \ 0 \le k < n \tag{4}$$

The filter w is defined as one minus the Hann window, see Figure 3.

$$w[j] = 1 - 0.5 \ (1 - cos(2\pi j/n)) \ , \ 0 \le j \le n \tag{5}$$

As a result, only the very low and high frequency components remain. In the time-power domain this corresponds respectively to a steady-state signal and step change.

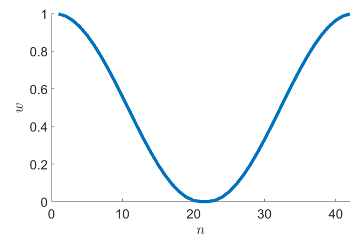


Fig. 3. The response of the filter w.

Because the relative difference in values of the components is more informative than the absolute difference, the frequency components are converted to a dB scale.

$$\hat{X}_{dB}[k] = 20 \ log_{10}(\hat{X}[k]) \tag{6}$$

These components are an informative indicator if there are any events present in the time window or not. This can be seen in Figure 2. It is clear that if an event is present, all the Cepstrum smoothed dB scaled frequency components have higher values (see Figure 4b) than when there is no event present (see Figure 4d). To detect if an event is present, it is checked if all frequency components are larger than a chosen threshold τ.

$$min(\hat{X}_{dB}[k]) > \tau \tag{7}$$

If yes, then this is labelled as an event. This threshold should however be trained, to obtain good detection ratios. How this is done, will be mentioned in next section.

The signal is processed by using overlapping time windows. If a window is

$$x_1 = \{p_{i+1}, p_{i+2}, ..., p_{i+n}\} \tag{8}$$

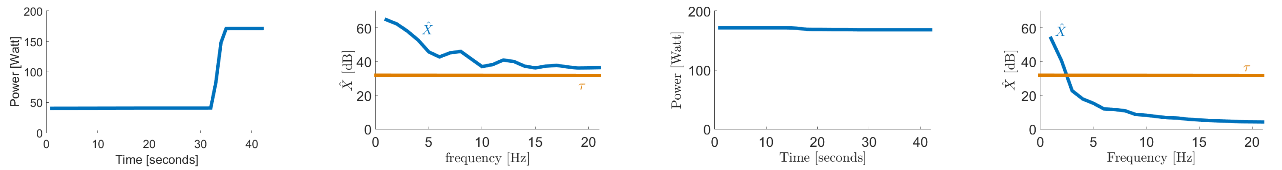
then the following window which overlaps it with three quarters, is represented by

$$x_2 = \{p_{i+n/4+1}, p_{i+n/4+2}, ..., p_{i+5n/4}\} \tag{9}$$

Note that this will cause a single event to be detected multiple times in consecutive time intervals. The exact timestamp of an events is pinpointed as the midpoint of the time interval covered by these windows. For an example, see Figure 4.

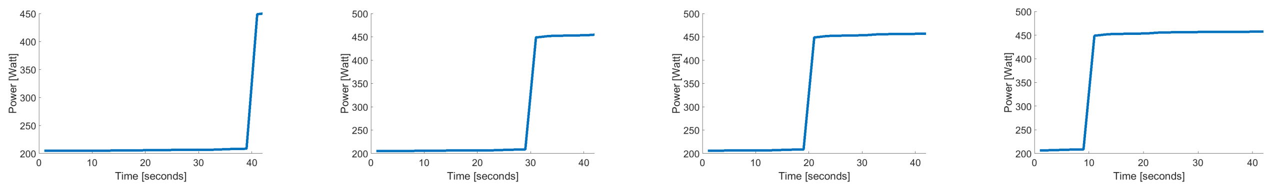
IV. Results

In this section the proposed method is compared with the χ^2 squared method by using the BLUED benchmark dataset [14]. From this data, the aggregated active power signal of 60Hz from a family residence in the United States for a whole week is considered. Every state transition of each appliance is labelled providing the ground truth. This power occurs in two phases, namely phase A and B. In total, 904 transitions are captured in phase A and 1578 in B. Each phase has its own properties, e.g. phase B is more noisy than phase A. For that reason, phase A and B are trained and tested separately.



(a) Power signal of an event. (b) Smoothed frequency components of an event. (c) Power signal of a non-event. (d) Smoothed frequency components of a non-event.

Fig. 2. Examples of windows with size $n = 40$ of a power trace and the corresponding smoothed frequency components $\hat{X}$ of an event, respectively 4a and 4b and a non-event, respectively 4c and 4d. In the smoothed frequency components a clear distinction is visible between an event and non-event.



(a) Window $\{p_1, ..., p_{40}\}$ (b) Window $\{p_{11}, ..., p_{50}\}$ (c) Window $\{p_{21}, ..., p_{60}\}$ (d) Window $\{p_{31}, ..., p_{70}\}$

Fig. 4. Four consecutive windows detecting the same event. Since the event detection is triggered four times in a row, the algorithm concludes that an event occurred in the center of the time interval covered by these windows.

To obtain the results, we split our data in five equal parts. One part is used for testing and the other four for training. On this training set we apply 5-fold cross-validation in order to avoid overfitting. From this cross-validation, optimal configuration settings for the methods are obtained and the evaluation on the test set gives us the final performance. This initial splitting is done 10 times so that the stability of our performance is proven. A schematic overview of this procedure is given in Figure 5. Note that for the division in parts, we take a file as a whole unit.

For the evaluation of the results, the advice given in [15] is followed. To assess the detection ratio's, the F-measure can be used which is defined as the harmonic mean of precision and recall.

$$F = 2 * \frac{precision * recall}{precision + recall} \tag{10}$$

$$precision = \frac{TP}{TP + FP} \tag{11}$$

$$recall = \frac{TP}{TP + FN} \tag{12}$$

where $precision$ is the fraction of detected events that are real and $recall$ is the fraction of true events that are detected, TP are the true-positives (correctly predicted events), FP are the false-positives (predicted events that were none), FN are the false-negatives (undetected events). The Formula's (10)-(12) depend on parameter settings of the algorithm. When the Cepstrum coefficients are used, the parameters are the window size $n = \{20, 40, 60, 80, 100\}$, the threshold $\tau = \{5, 10, 15, 20, 25, 30, 35, 40\}$. When the χ^2 method is used, the considered parameters are the window size $n = \{20, 40, 60, 80, 100\}$, the confidence level $\alpha = \{0.90, 0.95, 0.975, 0.99, 0.999\}$ and the window size $m = \{10, 30, 50, 70, 90, 110\}$ of the window used by the median filter as the signal is smoothed in the time domain [4].

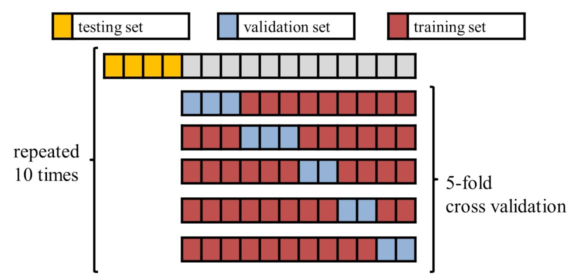


Fig. 5. A schematic overview of how training is done. The data is split in five equal parts. One part is used for testing and the other four parts are used for training. On the training set, 5-fold cross-validation is performed.

The experiments are repeated 10 times and the minimum, first quantile (Q1), median, third quantile (Q3) and maximum of the F-measure are reported in Table I. If all these values are close together, then this means that the algorithm is stable which is the case for both methods. Looking at the F-measure, it can be concluded that the Cepstrum analysis is as good as the χ^2 GOF statistic showing $\sim 98\%$ perfect event detection for phase A and $\sim 80\%$ for phase B.

Table I also describes the running time of the algorithms to process eight files. This is equal to the amount of files used for training in one fold. It can be seen that the Cepstrum method takes more time than the χ^2 method.

V. Conclusion

In this paper an event detection method is proposed that works in the frequency domain and uses Cepstrum smoothing

TABLE I
THE SPREAD OF THE F-MEASURE WHEN APPLYING THE χ^2 GOF METHOD AND CEPSTRUM ANALYSIS ON PHASE A AND B OF THE BLUED DATASET AND THE RUNNING TIME OF THE ALGORITHM TO PROCESS EIGHT DAYS.

	Phase A		Phase B	
	χ^2	Cepstrum	χ^2	Cepstrum
Min	86.2%	87.14%	68.89%	74.29%
Q1	98.06%	97.73%	76.5%	77.23%
Med	98.44%	98.01%	81.01%	80.035%
Q3	99.2%	98.4%	83.12%	81.84%
Max	99.74%	98.76%	85.1%	86.22%
Time	127 sec	926 sec	142 sec	1717 sec

for eliminating noise. With a reported F-measure equal to 98.01% for phase A and 80.03% for phase B on the BLUED dataset, the method is competing with the state-of-the-art. Although the method is more opaque than the traditional χ^2 method, a big advantage of this method is that the frequency components can be used for both event detection and appliance recognition. More details will be reported in a forthcoming paper.

REFERENCES

[1] Hart, George W., *Nonintrusive appliance load monitoring.*, Proceedings of the IEEE 80.12, 1992, 1870-1891.

[2] Wong, Yung Fei, et al., *Recent approaches to non-intrusive load monitoring techniques in residential settings.*, Computational Intelligence Applications In Smart Grid (CIASG), 2013 IEEE Symposium on. IEEE, 2013.

[3] Anderson, Kyle D., et al., *Event detection for non intrusive load monitoring.*, IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society. IEEE, 2012.

[4] Jin, Yuanwei, et al. *A time-frequency approach for event detection in non-intrusive load monitoring.*, SPIE Defense, Security, and Sensing. International Society for Optics and Photonics, 2011.

[5] Trung, Kien Nguyen, et al. , *Event Detection and Disaggregation Algorithms for NIALM System.*, the 2nd International Non-Intrusive Load Monitoring (NILM) Workshop. 2014.

[6] Volpi, Michele, et al., *Unsupervised change detection with kernels.*, Geoscience and Remote Sensing Letters, IEEE 9.6, 1026-1030, 2012.

[7] Luong, The Minh, et al., *Hidden Markov Model Applications in Change-Point Analysis.*, arXiv preprint arXiv:1212.1778, 2012.

[8] Grinblat, Guillermo L., et al., *Abrupt change detection with one-class time-adaptive support vector machines.*, Expert Systems with Applications 40.18. 7242-7249, 2013.

[9] Gu, Wanyi, et al., *Fast Change Point Detection for electricity market analysis.*, Big Data, 2013 IEEE International Conference on. IEEE, 2013.

[10] Borgert, B. P., M. J. Healy, and J. W. Tukey., *The quefrency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum and saphe craking.*, Proc. Symp. On Time Series Analysis, Rosenblatt, M. ed., 209-243, 1963.

[11] Kiran, PV Ravi,et al., *Application of Cepstrum in Passive Sonar.*, International Journal of Engineering Research and Applications (IJERA), 1919-1924 , 2012.

[12] Hirsch, Hans-Gnter, and David Pearce. *The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions.*, ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW), 2000.

[13] Kong, Seongbae, et al., *Home appliance load disaggregation using cepstrum-smoothing-based method.*, Consumer Electronics, IEEE Transactions on 61.1, 24-30, 2015.

[14] Anderson, Kyle, et al., *BLUED: A fully labeled public dataset for event-based non-intrusive load monitoring research*, Proceedings of the 2nd KDD workshop on data mining applications in sustainability (SustKDD), 2012.

[15] Makonin, Stephen, and Fred Popowich., *Nonintrusive load monitoring (NILM) performance evaluation.*, Energy Efficiency 8.4, 809-814, 2015.