An Event-Based Nonintrusive Load Monitoring Approach:
Using the Simplified Viterbi Algorithm

Nonintrusive load monitoring technologies are gaining popularity. The authors' proposed approach—a simplified event-detection algorithm based on maximum and minimum points, and a disaggregation method that applies a simplified Viterbi algorithm—saves time and works well for high-power appliances.

Tianqi Lu, Zhengguang Xu, and Benxiong Huang
Huazhong University of Science and Technology

Nonintrusive load monitoring (NILM), pioneered by George Hart,[1] refers to technologies aimed at taking aggregated energy-consumption data acquired from single-point measurements and separating it based on how it is consumed by individual appliances, without installing sensors for each appliance.[2,3] Most existing NILM methods can be divided into event-based and non-event-based.[3] The former generally consists of event detection and classification steps, so the accuracy of the event detection directly influences the results of the classification work. Commonly used event-detection algorithms include the cumulative sum detector,[4] the generalized likelihood ratio detector,[5] and the goodness-of-fit detector,[6] all three of which require a fixed window size for the reference set and current set. However, different electric appliances work in different styles, so this isn't optimal.

Non-event-based methods usually do not have an event-detection step; instead, they take every sample of aggregated power data into account. For these methods, the hidden Markov model (HMM) and its variants are widely used to model the appliance's consumption behavior.[3,7–10] Such methods usually use all sample points as data input, leading to large data input for long-term monitoring. Furthermore, because they consider all the combined states in calculations, the process becomes computationally expensive and time consuming as the number of combined states grows exponentially with the number of electrical appliances.

Recently, researchers have tried to reduce such computational complexity (see the "Related Work in Modeling Appliance Consumption Behavior" sidebar for a discussion of this work and how it differs from our approach). Here, we present an adaptive event-detection algorithm based on maximum and minimum points (MMP) without a fixed detection window. The proposed MMP detector searches the extreme points of the power curve and determines the duration of the power-on and power-off processes for different appliances adaptively. Meanwhile, the noise suppression operation is also used to improve detector performance. Then, we exploit a modified version of a *factorial HMM* (FHMM), which considers both the

Published by the IEEE CS ■ 1536-1268/17/$33.00 © 2017 IEEE

Related Work in Modeling Appliance Consumption Behavior

Recently, some researchers have tried to reduce the computational complexity of modeling appliance consumption behavior. Matthew Johnson and Alan Willsky used the factorial variant of a hidden semi-Markov model and reduced the number of potential change points that need to be considered by simply computing first differences and thresholding.[1] However, details about the event-detection algorithm were not presented. In addition, they hand-picked only five appliances (with quite different power levels) for testing and evaluation.

Michael Zeifman and Kurt Roth proposed using a Viterbi algorithm with Sparse Transitions (VAST) on multiple transition matrices.[2] They assumed that two or more appliances don't start or stop simultaneously, but analyzing low-frequency data in a real house showed that it's very likely for two appliances to change states simultaneously.

Stephen Makonin and his colleagues took advantage of sparsity in matrix storage and processing and presented a sparse Viterbi algorithm based on matrix sparsity.[3]

Different from Johnson and Willsky,[1] we describe an event detection with noise suppression operation in detail, and we only use the two power sequences without considering the complex time duration facts of appliances. More appliances in a real house are used in this article to evaluate the proposed approach. In particular, we present a simplified Viterbi algorithm. Like Zeifman and Roth,[2] we exploit the sparsity of transitions between appliances' states to decompose the main algorithm. Differently, we let two appliances change states at the same time, which is closer to reality, and we simplify the state transitions by setting two power thresholds instead of limiting the number of appliances with similar power levels (such as using triples of loads or load pairs to avoid one large sparse transitions matrix). This is because there can be more than three appliances whose power levels are very close in a real house, and power fluctuation in the system can affect the possible loads you estimate if you're limited to just load triples or pairs.

Makonin and his colleagues exploited the sparsity matrix by pointing that some states transitions would not make much sense.[3] Unlike them, the work in this article sets two power thresholds to restrict the possible states at the current moment without prior knowledge of the relations among all states. The simplified Viterbi algorithm proposed here could also largely decrease the dimensionality of state space.

REFERENCES

1. M.J. Johnson and A.S. Willsky, "Bayesian Nonparametric Hidden Semi-Markov Models," *J. Machine Learning Research*, vol. 14, no. 1, 2013, pp. 673–701.
2. M. Zeifman and K. Roth, "Viterbi Algorithm with Sparse Transitions (VAST) for Nonintrusive Load Monitoring," *Proc. IEEE Computational Intelligence Applications in Smart Grid* (CIASG), 2011, pp. 1–8.
3. S. Makonin et al., "Exploiting HMM Sparsity to Perform Online Real-Time Nonintrusive Load Monitoring," *IEEE Trans. Smart Grid*, 2015, pp. 1–11.

aggregated and the differential power data.[9] We call this model the *Additive and Difference FHMM* (ADFHMM). To reduce complexity, we combine the ADFHMM with the MMP event detector and present a simplified Viterbi algorithm, which could help decrease the dimensionality of the state space.

Event Detection

Not all appliances work in the same way, so it's difficult to set the window size of the reference set. State changes of appliances are accompanied by power changes and often occur at the maximum or minimum points of the total power curve. There are only two parameters in our algorithm: the power threshold G_p and the noise threshold G_n. G_p can be calculated from the variation of the noise, and G_n can be set by users. For example, if we want to look for events with power changes of more than 50W, we can set $G_p = 50$. Supposing that the total power sequence is denoted as $\mathbf{x} = x_1, \ldots, x_T$, the algorithm follows three steps as follows.

Step 1 is to search for extreme points. The maximum points satisfy the following:

$$\begin{cases} x_n \geq x_{n-1} \\ x_n \geq x_{n+1} \\ x_n \neq \dfrac{x_{n-1} + x_{n+1}}{2} \end{cases} \quad (1)$$

and the minimum points satisfy

$$\begin{cases} x_n \leq x_{n-1} \\ x_n \leq x_{n+1} \\ x_n \neq \dfrac{x_{n-1} + x_{n+1}}{2}. \end{cases} \quad (2)$$

Step 2 is to determine the rising and falling intervals according to the extreme points. Suppose x_i, x_j, and x_k ($i < j < k$) are three adjacent extreme points. Because the minimum point and maximum point must be alternating, we define x_i and x_k as minimum points and x_j as the maximum point. If $x_j - x_i > G_p$, then we define $R^+_{i,j} = \{x_n \mid i \leq n \leq j\}$ as the rising interval. Similarly, if $x_j - x_k > G_p$, then we

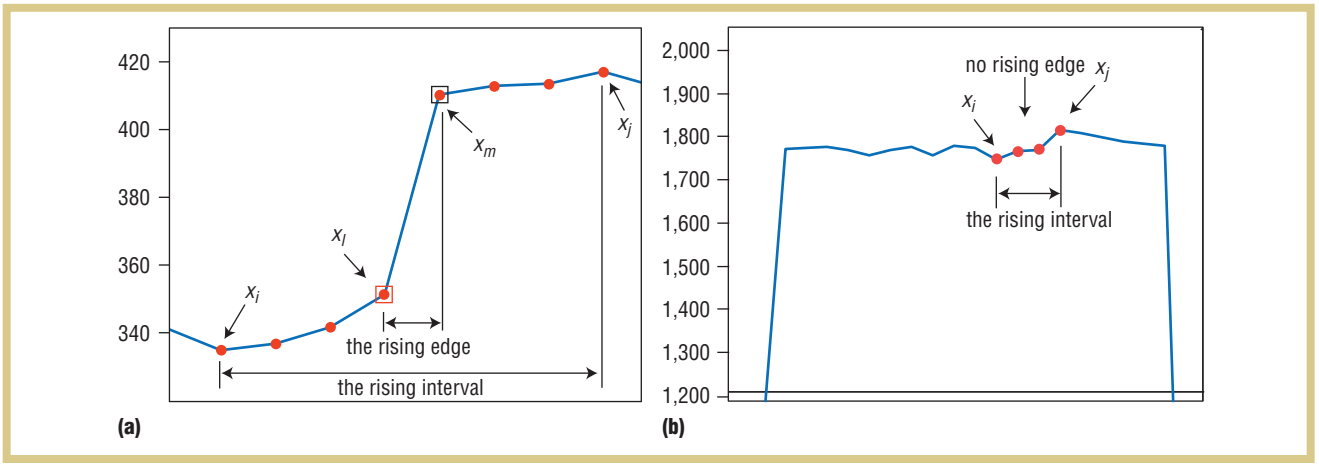


Figure 1. Edge searching in the rising interval: (a) the case for the real power switch, and (b) the case for the slow power change.

define $R^{-}_{j,k} = \{x_n \mid j \le n \le k\}$ as the falling interval.

Step 3 is to search the rising and falling edges from the rising and falling intervals. The search process of the rising edge is to find adjacent points of which the power difference is greater than the noise threshold G_n:

- *Forward search*: Searching from $t = i$ to $t = j - 1$, if $x_{t+1} - x_t > G_n$, the search stops and x_t is the start point of the rising edge. If the point meeting the condition is not found, the search fails.
- *Backward search*: Searching from $t = j$ to $t = i + 1$, if $x_t - x_{t-1} > G_n$, the search stops and x_t is the end point of the rising edge. If the point meeting the condition is not found, the search fails.

If both of the searches succeed, we denote the start point of the rising edge as x_l and the end point as x_m. If $x_m - x_l > G_p$, we denote the rising edge as $E^{+}_{l,m} = \{x_t \mid l \le t \le m\}$. Similarly, we denote the falling edge as $E^{-}_{l,m} = \{x_t \mid l \le t \le m\}$. If any of the searches fail, no event is detected. In this case, the rising or falling intervals might be caused by the slow change of the application power.

In the proposed algorithm, Step 1 determines the potential event interval, Step 2 determines whether an event arises by the power threshold, and Step 3 eliminates the noise interference and small power changes using the noise threshold.

Figure 1 shows two examples, where $G_p = 50$ and $G_n = 20$. The forward search is done from x_i, while the backward search is done from the point x_j. Figure 1a shows that the state of the appliance has changed, so the rising edge can be found from the rising interval. However, in the case presented in Figure 1b, the rising interval is caused by the internal power change of appliances, rather than the power switch. The power difference between the maximum and minimum points is larger than G_p, but the power difference between every two neighboring points in the interval is less than G_n, so the MMP detector accurately determines that there is no event in this interval.

Generally speaking, there is a steady stage between two adjacent events. So supposing that L steady stages are obtained after the event detection, we calculate the mean and the differential power of these steady stages and take these values as the data input of the next step. The mean aggregated power sequence is defined as $\mathbf{y} = y_1, y_2, \ldots, y_L$. The differential power sequence is defined as $\Delta\mathbf{y} = \Delta y_1, \Delta y_2, \ldots, \Delta y_L$, where

$$\Delta y_l = \begin{cases} 0, & l = 1 \\ y_l - y_{l-1}, & l = 2, 3, \ldots, L. \end{cases}$$

Appliance Models

We use the ADFHMM shown in Figure 2 to model appliance behavior. We consider both the mean and differential power sequences calculated earlier. Table 1 defines some of the notations.

Supposing that there are N appliances, and all appliances have only two states, in the ADFHMM, we define $S = \{S_1, S_2, \ldots, S_K\}$ as the hidden combined states (or super-states) set of all appliances, and $K = 2^N$ is the number of combined states. We define the observed power at steady stages as $O_l = (y_l, \Delta y_l)$. Then, the ADFHMM is defined by the set of parameters $\lambda = \{\boldsymbol{\pi}, \mathbf{A}, \mathbf{B}\}$, and each parameter is defined as follows:

- the initial probability $\boldsymbol{\pi}$: $\pi_k = \mathrm{P}(\mathbf{q}_1 = S_k)$, $k = 1, 2, \ldots, K$.
- the transition matrix $\mathbf{A}$: $\mathbf{A} = \{a_{jk}, 1 \le j, k \le K\}$.
- the emission matrix $\mathbf{B}$: $\mathbf{B} = \{b_k(O_l), 1 \le l \le L, 1 \le k \le K\}$.

To facilitate the calculation, we take the logarithm of the probability. For the

transition probability and initial probability, we use the uniform distribution to describe their probability density function. For example, for K combined states, we set $a_{jk} = 1/K$ and $\pi_k = 1/K$, and this performs well when using our disaggregation algorithm. There are several reasons for setting the two parameters.

First, the disaggregation method we propose here mainly depends on the event detection and power threshold to simplify complexity. The transition probability shares a bit smaller proportion during the calculation process. Second, the exact transition probability could be obtained only through a long period of training and learning or collections of priors, which is time-consuming. To test and verify, we use both the simple $1/K$ and the exact transition probability matrix. The accuracy of the former is within the acceptable range and not much lower than the latter. Therefore, the simple solution to transition probability described here is practical and saves time. Additionally, we assume that the power consumption of each appliance follows the Gauss distribution, and each is independent.

The Viterbi Algorithm

Next, we estimate the hidden states $\mathbf{q}^*$.[10] More formally, we try to solve this decoding problem:

$$\mathbf{q}^* = \arg\max_{\mathbf{q}} \log\left(P(\mathbf{q} \mid \mathbf{O}, \lambda)\right). \tag{3}$$

Applying the Bayes' theorem, we can simplify this probability:

$$P(\mathbf{q} \mid \mathbf{O}, \lambda) = \frac{P(\mathbf{O} \mid \mathbf{q}, \lambda)P(\mathbf{q})}{P(\mathbf{O})}. \tag{4}$$

Because the denominator has nothing to do with $\mathbf{q}$, we can write the problem in Equation 3 as follows:

$$\mathbf{q}^* = \arg\max_{\mathbf{q}} \log \left(P(\mathbf{q}_1) \prod_{l=2}^{L} \left(P(\mathbf{q}_1 \mid \mathbf{q}_{l-1}) P\left(O_l \mid \mathbf{q}_l\right)\right) \right). \tag{5}$$

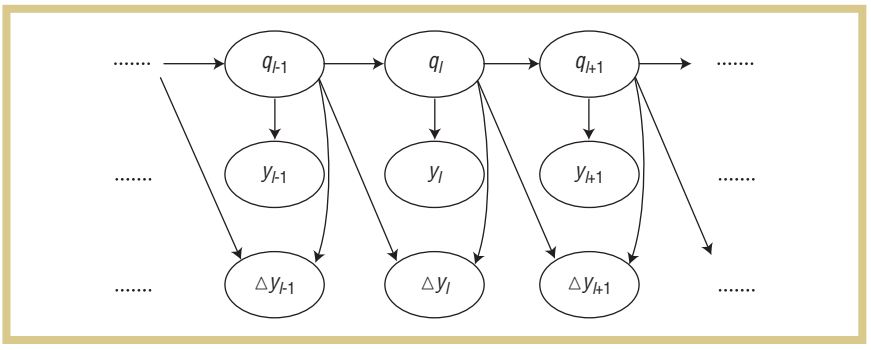


Figure 2. The Additive and Difference Factorial Hidden Markov Model (ADFHMM) that we use to model appliance behavior.

TABLE 1
Specific definition of notations.

Notations	Definition
T	Number of sampling points
L	Number of steady stages
N	Number of appliances
K	Number of combined states, $K = 2^N$
y_l	Actual value of aggregated power
Δy_l	Differential power, $\Delta y_l = y_l - \Delta y_{l-1}$
C_{th}	Combined power threshold
D_{th}	Differential power threshold
$\boldsymbol{O}$	Observed sequence (composed of y_l and Δy_l)
$\boldsymbol{S}_k$	The k-th state of the state space $\mathbf{S}$
$\boldsymbol{q}_l$	Combined state of steady stage l
$q_l^{(i)}$	State of appliance i at steady stage l, $q_l^{(i)} \in \{0, 1\}$
u_i	Power level of appliance i
y_l^*	Estimation of aggregated power
$Q(\mathbf{q}_{l-1}, \mathbf{q}_l)_{j,k}$	$Q(\mathbf{q}_{l-1}, \mathbf{q}_l)_{j,k} \in \{0, 1\}$, $Q(\mathbf{q}_{l-1}, \mathbf{q}_l)_{j,k} = 1$ represents $\mathbf{q}_{l-1} = \mathbf{S}_j$, $\mathbf{q}_{l-1} = \mathbf{S}_k$

The Viterbi algorithm aims to solve this in the shortest way possible and tries to find the traversed node according to the shortest path. So the objective function of the Viterbi algorithm can be written as

$$\max \log\left(p\left(L, \mathbf{q}_L\right)\right).$$

In the ADFHMM, we exploit both the aggregated power and the difference power, so the logarithmic form of the probability of the combined state $\mathbf{q}_l$ at stage l can be defined as

$$\log\left(p\left(l, \mathbf{q}_l\right)\right) = \begin{cases} \log\left(b_k(O_1)\right), & l = 1, \\ \log\left(p\left(l-1, \mathbf{q}_{l-1}\right) a_{j,k} b_k(O_l)\right), & l > 1. \end{cases} \tag{6}$$

Then we use Φ to define the objective function. The two binary variables, $Q(q_l^{(i)})$ and $Q(q_{l-1}^{(i)}, q_l^{(i)})$, represent the state and the transition state of load i at

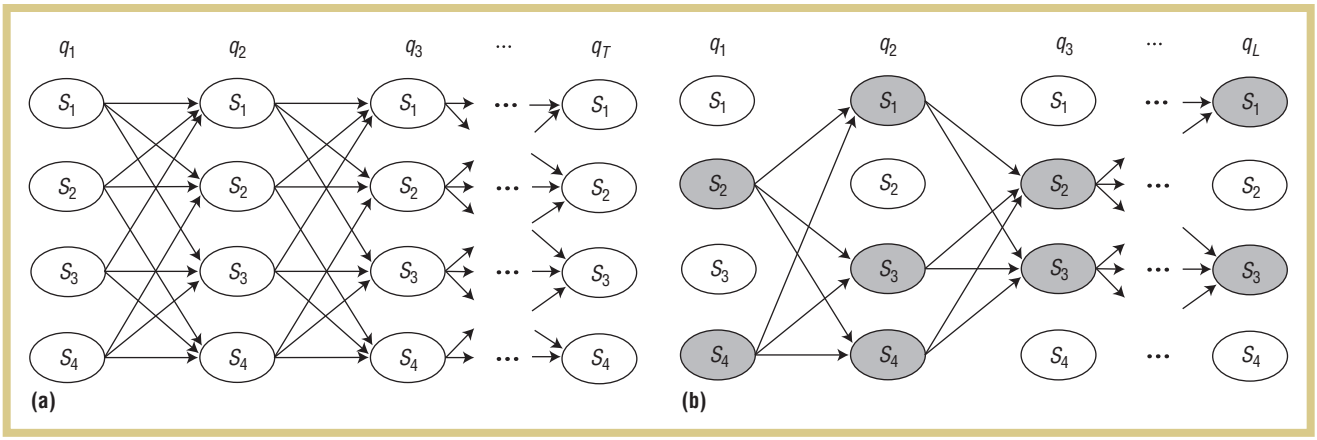


Figure 3. The trellis of (a) the traditional Viterbi algorithm and (b) the simplified Viterbi algorithm.

stage l, respectively. Then, the objective function can be rewritten as

$$\max \Phi\left(Q(q_l^{(i)})_j, Q(q_{l-1}^{(i)}, q_l^{(i)})_{j,k}\right) \quad (7)$$
$$= \max \begin{pmatrix} -\frac{1}{2}\sum_{l=1}^{L}\left\| y_l - \sum_{i,j} u_j^{(i)} Q(q_l^{(i)})_j \right\|^2 - \\ \frac{1}{2}\sum_{l=2}^{L}\left\| \Delta y_l - \sum_{i,j} \Delta u_{j,k}^{(i)} Q(q_{l-1}^{(i)}, q_l^{(i)})_{j,k} \right\|^2 + \\ \sum_{t=2,i,j,k} Q(q_{l-1}^{(i)}, q_l^{(i)})_{j,k} \log(P_{j,k}^{(i)}) \end{pmatrix}.$$

In Equation 7, $u_j^{(i)}$ is the mean power level of load i at state j, and $\Delta u_{j,k}^{(i)}$ is the difference mean power of load i that transfers from state j to k. We assume that the power of load follows Gaussian distribution. In our work, we mainly rely on mean power so that the variance is set to the constant.

The Traditional Viterbi Algorithm

To solve this decoding problem, we often use the Viterbi algorithm. For the traditional Viterbi, factors of the appliance model ADFHMM and all aggregated power data measured are used as the input. Many researchers use an additional constraint—at most one appliance can change states at a time.

We use two appliances as an example, and the aggregated power sequence has T sampling points. Figure 3a shows the trellis graph of traditional Viterbi, illustrating that there are approximately $2^N\left(C_N^1 + C_N^0\right)$ paths to calculate each time. When there are many appliances, this computation would largely increase.

The Simplified Viterbi Algorithm

In our simplified Viterbi algorithm, first we exploit the result of event detection and take the mean aggregated power sequence, $\mathbf{y}$, and the differential power sequence, $\Delta\mathbf{y}$, as the inputs. Consequently, every input is caused by the state changes of appliances, so $Q(q_{l-1}, q_l)_{j,k} = 0$ when $j = k$.

Second, by observing the events of "House 1" in the Reference Energy Disaggregation Dataset (REDD),[11] we find that the most common situation is for one device to change states, although sometimes two devices change at the same time. It's almost impossible for three or more devices to change simultaneously. So we allow at most two appliances to change states at a time. (Statistically, the frequency of one device changing states at a time is approximately 50 times higher than that of two devices changing simultaneously.) If we use Z as the number of appliances that change states at a time, then we set a parameter

$$\alpha = \begin{cases} 1, & Z = 1 \\ 0.02, & Z = 2 \\ 0, & Z = 0, Z > 2 \end{cases}, \quad (8)$$

and we calculate the transition probability as follows:

$$P(\mathbf{q}_l = \mathbf{S}_k \mid \mathbf{q}_{l-1} = \mathbf{S}_j) = \alpha \cdot a_{jk} Q(\mathbf{q}_{l-1}, \mathbf{q}_l)_{j,k}. \quad (9)$$

Within the constraint described in Equation 8, we must calculate $2^N\left(C_N^2 + C_N^1\right)$ transitions each time to find the most likely path. With the increase of N, the computation would be larger than the traditional Viterbi algorithm. So we include additional constraints to simplify the state space.

Assume that power levels of all N appliances are given, defined as $\mathbf{u} = (u_1, u_2, \ldots, u_N)$. So for a combined state at stage l, $\mathbf{q}_l = \left\{q_l^{(1)}, q_l^{(2)}, \ldots, q_l^{(N)}\right\}$. $q_l^{(i)} = 1$ represents the ON state of appliance i and $q_l^{(i)} = 0$ represents the OFF state. Then we can estimate the aggregated power,

$$y_l^* = \sum_{i=1}^{N} q_l^{(i)} u_i. \quad (10)$$

Of course, we can also calculate the differential aggregated power between $\mathbf{q}_l$ and $\mathbf{q}_{l-1}$,

$$\Delta y_l^* = \begin{cases} 0, & l = 1 \\ y_l^* - y_{l-1}^*, & 2 \le l \le L \end{cases}. \quad (11)$$

We use the combined power threshold C_{th} and the differential power

TABLE 2
The comparison results.

Device	Power level (W)	Our approach			The control group			AFAMAP*		
		F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall
light2	54	0.5840	0.4389	0.8727	0.7199	0.6075	0.8833	0.5831	0.4399	0.8643
light1	63	0.8177	0.7187	0.9484	0.8084	0.7033	0.9505	0.7733	0.6501	0.9543
kOutlet3	72	0.5012	0.4646	0.544	0.5318	0.5297	0.5339	0.5419	0.5945	0.4978
light3	80	0.6933	0.5601	0.9096	0.4767	0.3300	0.8576	0.6632	0.5147	0.9322
refrigerator	193	0.9352	0.8946	0.9797	0.9457	0.9267	0.9655	0.9076	0.8622	0.9580
washerdryer1	466	0.8271	0.7145	0.9817	0.8531	0.7582	0.9751	0.9073	0.8438	0.9811
dishwasher	669	0.5905	0.4768	0.7754	0.602	0.4985	0.7598	0.2164	0.1283	0.6913
kOutlet1	1076	0.9898	0.9875	0.9922	0.9549	0.9432	0.9668	0.8928	0.8365	0.9572
microwave	1527	0.6863	0.5229	0.9985	0.7955	0.7058	0.9114	0.8429	0.7321	0.9931
kOutlet2	1535	0.8753	0.7806	0.9962	0.943	0.909	0.9796	0.6768	0.5140	0.9903
bathroomgfi	1605	0.9691	0.9443	0.9953	0.9472	0.9538	0.9407	0.9655	0.9394	0.9932
washerdryer2	2711	0.9429	0.8920	1	0.9451	0.8958	0.9999	0.9836	0.9677	1.0000
oven	4000	0.8999	0.8181	0.9999	0.9726	0.947	0.9997	0.8953	0.8106	0.9998

*Additive Factorial Approximate Maximum A Posteriori.

threshold D_{th} to constrain the transition paths of the state space. We define the transition set as $\mathbf{Q}_l$, $2 \le l \le L$, where

$$\mathbf{Q}_l = \left\{ (\mathbf{q}_{l-1}, \mathbf{q}_l) \middle| \begin{Bmatrix} \left| y_l^* - y_l \right| < C_{th} \\ \left| \left| \Delta y_{l-1}^* \right| - \left| \Delta y_{l-1} \right| \right| < D_{th} \\ \mathbf{q}_l, \mathbf{q}_{l-1} \in \mathbf{S}, \mathbf{q}_l \neq \mathbf{q}_{l-1} \end{Bmatrix} \right\}. \quad (12)$$

In addition, we have

$$Q(\mathbf{q}_{l-1}, \mathbf{q}_l) = \begin{cases} 1, & (\mathbf{q}_{l-1}, \mathbf{q}_l) \in \mathbf{Q}_l \\ 0, & (\mathbf{q}_{l-1}, \mathbf{q}_l) \notin \mathbf{Q}_l \end{cases}. \quad (13)$$

Figure 3b illustrates an example, showing that fewer calculations are required for each transition compared to the traditional algorithm, because

- the simplified algorithm includes the results of event detection, greatly reducing the amount of input;
- we allow at most two appliances to change states simultaneously; and
- we simplify the state space by setting two power thresholds.

Consequently, we greatly decrease the computational complexity of the simplified algorithm.

Experiments and Results

In our experiments, we used REDD to test our algorithm, because it contains both aggregated and circuit-level power data. Also, to make comparisons, here we use a two-week-long power sequence of House 1 in REDD as an example. The number of appliances and their power levels appear elsewhere.[12]

For binary classifications, there are four possible results: true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*). We define

$$precision = \frac{TP}{TP + FP}, \quad recall = \frac{TP}{TP + FN}$$

and

$$F\text{-}measure = \frac{2precison \times recall}{precision + recall}.$$

In keeping with original evaluations,[12] we also use $\rho = 0.2$ and $\theta = 30$. As for multistate appliances, we calculate the mean power level, assuming that each appliance has only two states.

We also conducted several experiments regarding the power threshold settings, C_{th} and D_{th}, and concluded that we can set them according to the load environment. We set C_{th} to the mean value of power levels of all loads, and we set the parameter D_{th} dynamically. These parameter settings could apply to other datasets.

First, we sorted the power levels of all loads in ascending order, so $u_i \le u_{i+1}$. Then,

$$C_{th} = \frac{1}{N} \sum_{i=1}^{N} u_i, \quad (14)$$

$$D_{th} = \begin{cases} 0.6 \times \frac{1}{N-1} \sum_{i=1}^{N-1} \left| u_{i+1} - u_i \right|, & \left| \Delta y \right| < C_{th}, \\ 0.6 \times \left| \Delta y \right|, & \left| \Delta y \right| \ge C_{th}. \end{cases} \quad (15)$$

Compared to the control group, which had no threshold, our approach's

TABLE 3
The average results of all houses in the Reference Energy Disaggregation Dataset (REDD).

	Our approach			AFAMAP		
House	F-measure	Precision	Recall	F-measure	Precision	Recall
1	0.7919	0.7088	0.9226	0.7082	0.7995	0.7376
2	0.7775	0.7272	0.9057	0.6452	0.5921	0.7549
3	0.8091	0.7856	0.8606	0.6859	0.5779	0.8677
4	0.7403	0.6461	0.8776	0.7345	0.6799	0.8541
5	0.8067	0.7716	0.8824	0.7718	0.704	0.9361
6	0.8911	0.8458	0.9607	0.7577	0.6795	0.9087
Mean	0.8028	0.7475	0.9016	0.7172	0.6722	0.8432

Our proposed MMP detector has fewer parameters but is still robust, and it's independent of the detection window.

accuracy is only a little lower, yet it required much less time (see Table 2).

We also compared the ADFHMM to the Additive Factorial Approximate Maximum A Posteriori (AFAMAP) algorithm,[9] which also considers both the aggregated and difference power and includes a generic component that can take on arbitrary values to solve unmodelled observations. If all of the load models are known, the generic component is not necessary. If we get rid of the generic component, the optimization problem of the AFAMAP algorithm[9] could be strikingly similar to the objective function of the Viterbi algorithm.[7]

So we also compared our approach and the AFAMAP algorithm after removing the generic component. The AFAMAP algorithm is non-event-based and constrains the posterior to allow at most one hidden state to change at a time. In our work, we combine the model inference with accurate event detection, which uses the noise suppression operation. In particular, we exploited the mean power of steady stages after event detection and allowed two appliances to change states simultaneously, improving precision (see Table 2).

The proposed approach performed better than AFAMAP for most appliances. For high-power devices, such as "oven," "washerdryer2," "bathroomgfi," "kOutlet1," and "kOutlet2," the F-measure of our approach was more than 0.8—sometimes even more than 0.9. For some low-power devices, such as "light1," "light2," and "light3," the F-measure could be lower but was still better than the AFAMAP. For "refrigerator," which is frequently used and has a steady power level, we accurately estimated its behavior. Compared with results presented elsewhere,[12] the AFAMAP here performed better due to the fact that we got rid of the generic component under the premise that all of the load models were known.

We also applied our approach to all houses in REDD and compared it with the AFAMAP algorithm. Table 3 presents the average results. For each house, we selected several loads that were used frequently or that had high power levels during that two-week time period. For instance, in House 1, four devices were seldom used, so we didn't consider them in the decomposition work. In House 2, the power of the washer and dryer was almost lower than 10W, so we only included the other eight devices. Similarly, we considered mainly used loads in all other houses using the same principle. Table 3 shows that the proposed approach performed better for all houses in REDD.

For the traditional Viterbi without using event detection, the computational complexity is $O(K^2T)$. In the proposed approach, we applied event detection to the raw sampling points of aggregated power data, potentially reducing the complexity to $O(K^2L)$. On the one hand, we largely decreased the amount of input data. On the other hand, using the mean power of steady stages can suppress the noise to some extent. In our experiments, we used two-week-long data with $T = 332{,}711$ sampling points, but only $L = 3{,}002$ steady stages had to be considered after event detection.

Furthermore, we allowed at most two appliances to change states simultaneously, which is more accordant with the real world. This makes our

disaggregation results more accurate. To further reduce the computation, we set two power thresholds to simplify the state space. We considered $N = 13$ loads in the experiment. For the traditional Viterbi algorithm, we have to calculate $2^N\left(C_N^1 + C_N^0\right) = 114{,}688$ transition paths each time. For the proposed simplified Viterbi, we only had to calculate approximately 4,182 transition paths, which is roughly 1/27th of the front. Thus we could disaggregate aggregated power much faster than with the traditional Viterbi but still with high disaggregation accuracy.

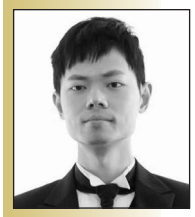
Our proposed MMP detector has fewer parameters but is still robust, and it's independent of the detection window. Moreover, the results of our experiments reveal that our proposed approach saves time and works especially well for high-power appliances. In the future, we will test our algorithm in other publicly available datasets and will try to implement the algorithm in a real-time system. P

REFERENCES

1. G. Hart, "Nonintrusive Appliance Load Monitoring," *Proc. IEEE*, vol. 80, no. 12, 1992, pp. 1870–1891.
2. M. Zeifman and K. Roth, "Nonintrusive Appliance Load Monitoring: Review and Outlook," *IEEE Trans. Consumer Electronics*, vol. 57, no. 1, 2011, pp. 76–84.
3. Y.F. Wong et al., "Recent Approaches to Non-Intrusive Load Monitoring Techniques in Residential Settings," *IEEE Symp. Computational Intelligence Applications in Smart Grid* (CIASG), 2013, pp. 73–79.
4. S.R. Mohanty, A.K. Pradhan, and A. Routray, "A Cumulative Sum-Based Fault Detector for Power System Relaying Application," *IEEE Trans. Power Delivery*, vol. 23, no. 1, 2008, pp. 79–86.
5. K.D. Anderson et al., "Event Detection for Non Intrusive Load Monitoring," *Proc. Ann. Conf. IEEE Industrial Electronics Soc.*, 2012, pp. 3312–3317.
6. Y. Jin et al., "Robust Adaptive Event Detection in Non-Intrusive Load Monitoring for Energy Aware Smart Facilities," *Proc. Conf. Acoustics, Speech and Signal Processing*, 2011, pp. 4340–4343.
7. R. Bonfigli et al., "Unsupervised Algorithms for Non-Intrusive Load Monitoring: An Up-To-Date Overview," *Proc. 15th Int'l Conf. Environment and Electrical Engineering* (EEEIC), 2015, pp. 1175–1180.
8. O. Parson et al., "Using Hidden Markov Models for Iterative Non-Intrusive Appliance Monitoring," *Neural Information Processing Systems Workshop on Machine Learning for Sustainability*, 2011; https://eprints.soton.ac.uk/272990/1/MLSUST2011.pdf.
9. J.Z. Kolter and T. Jaakkola, "Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation," *Proc. Int'l Conf. Artificial Intelligence and Statistics*, 2012, pp. 1472–1482.
10. H. Kim et al., "Unsupervised Disaggregation of Low Frequency Power Measurements," *Proc. 2011 SIAM Int'l Conf. Data Mining*, vol. 11, 2011, pp. 747–758.
11. J.Z. Kolter and M.J. Johnson, "REDD: A Public Data Set for Energy Disaggregation Research," *Proc. SustKDD Workshop on Data Mining Applications in Sustainability*, 2011, pp. 59–62.
12. H. Shao, M. Marwah, and N. Ramakrishnan, "A Temporal Motif Mining Approach to Unsupervised Energy Disaggregation: Applications to Residential and Commercial Buildings," *Proc. 27th AAAI Conf. Artificial Intelligence* (AAAI), 2013, pp. 1327–1333.

the AUTHORS

Tianqi Lu is a student of electronic information and communications at Huazhong University of Science and Technology, Wuhan, China. Her research focuses on signal processing in power systems. Lu received her MS in communication and information systems from Huazhong University of Science and Technology. Contact her at 573904089@qq.com.



Zhengguang Xu is a lecturer of electronic information and communications at Huazhong University of Science and Technology. His research interests include signal analysis for communication systems and power systems. Xu received a PhD in communication engineering from Huazhong University of Science and Technology. Contact him at xray@hust.edu.cn.

Benxiong Huang is a professor of electronic information and communications at Huazhong University of Science and Technology. His research interests include power systems and system performance. Huang received his PhD in communication engineering from Huazhong University of Science and Technology. Contact him at huangbx@hust.edu.cn.