TUDELFT

MASTER THESIS

Non-Intrusive Appliance Load Monitoring using the Viterbi Algorithm (NIALM-VA)

Author: *Samira Bahrami*

Supervisors:Technical University of Delft:
Prof.Geert Leus
Philips Research:
Dr.Ronald Rietman
Ying Wang
Silvia Bertagna de marchi

Philips
EWI

May 2014

“ ”

Samira Bahrami.
Delft, The Netherlands

TUDELFT

Abstract

EWI
Circuits and Systems

Master of science

Non-Intrusive Appliance Load Monitoring using the Viterbi Algorithm (NIALM-VA)

by Samira Bahrami

The goal of Non-Intrusive Appliance Load Monitoring (NIALM), or energy disaggregation, is to deduce which devices are active and how much energy they consume from observation of the time evolution of the total voltage or current in the electrical network. In this thesis, energy disaggregation is performed from the time series of power meter readings, by making use of the fact that different types of appliance can be distinguished by the statistical properties of their signatures, i.e., power consumption behaviour over time. Optimal detectors, using the Viterbi algorithm, are derived for three types of signature: 1) only power levels, no time information, 2) power levels with exponentially distributed lifetimes and 3) power levels with Gaussian lifetimes with given means and variances. The detectors have been implemented in MATLAB, and their performance is compared for various inputs.

Delft University of Technology
Department of
EWI

The undersigned hereby certify that they have read and recommend to the Faculty of EWI for acceptance a thesis entitled **'Non-Intrusive Appliance Load Monitoring using the Viterbi Algorithm (NIALM-VA)'** by **Samira Bahrami** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated:

Chairman: _____

Advisors: _____

_____

Committee Members: _____

_____

_____

Contents

List of Figures

List of Tables

Chapter 1

Introduction

People today are more interested to achieve information about energy disaggregation for different purposes, mostly for energy efficiency and economical reasons. Energy disaggregation is the task of taking a whole-home energy signal and separating it into its component appliance, it also called non-intrusive appliance load monitoring[2]. Awareness of the detailed consumption of energy can positively encourage the people to reduce their unneeded usage of power. It also provides companies the opportunity to manage their system energy consumptions in the most efficient way. This will lead to a large amount of energy conservation.

The other interesting application of energy disaggregation, is related to security. By knowing the individual appliance load behavior, it is possible to detect an unexpected event and send an alarm when the power usage passes certain threshold, which could be due to system damage or intrusion of the system, house or company.

Appliance load monitoring also provides the opportunity to monitor people's behavior in different places. This is valuable in the case of monitoring the behavior of aged people who live alone, to providing them some facilities if they are not able to do their routines anymore. This kind of information can be a concern of some other human science researches.

Non-Intrusive Appliance Load Monitoring (NIALM) in contrast to Intrusive Appliance Load Monitoring (IALM) does not require to place a sensor on every individual appliance to monitor its behaviour. It provides the possibility to derive the required information from the current and voltage of the total load. NIALM requires more complex software but simpler hardware than IALM. Furthermore NIALM is cheaper and easier to setup than IALM, and thus much more desirable.

There are two major possibilities to approach NIALM. The first one is through analysing the current, which is sampled at a high frequency of about 10 kHz. The other one, which is the concern of this research, is by analysing the power consumption with a low sampling frequency rate of 1 Hz due to the fact that the process of calculating the average power from voltage and current in a period of time makes this sampling rate slow. But because of the presence of a power-meter in any home, it is desirable to find a way to get the best solution from power consumption information.

From a certain point of view, the power consumption data contains two kinds of information. First, the power value at each time sample, which is the sum of the power signatures of the appliances which are on at that point, the jump between two power samples could be caused by simultaneous events. Considering that each appliance has a certain signature with a time dependent pattern, time is the second given information. There is a possibility to derive a lot of information from event times, which is one of the concerns of this research.

To analyse the power consumption with the aim of finding whether the appliances are on or off during the observed time, the signature of each appliance is required. This could be obtained by a manual setup (MS-NIALM) or an automatic setup (AS-NIALM). At first, we will start by MS-NIALM, and improve this algorithm to AS-NIALM could be the next step.

Each appliance has a unique feature in power consumption behaviour, which is called an appliance signature. Appliances from different brands or versions may have slightly different signatures. Also the human behaviour can make some change in the signature of an appliance. For example, refrigerator signature can change during the certain period of time because of that its door was opened for a while. By monitoring the appliance signatures for a long period of time, some information based on time of usage and probability of being on or off in a certain period of time is achieved. Proposed models here are based on these stochastic models, which are obtained from these sorts of data. Therefore depends on how much our signatures database is rich based on the number of the appliances and measurement time, detection can have a different level of accuracy.

Researches on power signatures done so far mostly tried to build an appliance models based on clustering, then finding the appliance's behaviour, putting them in statistic tabulate and naming the appliances by signal processing techniques. There are some difficulties to distinguish two different appliances when they have some similarity in their sequence of power jumps using these algorithms. Time

is an important information which is not considered in most of these researches. There is no such a rich algorithm available which can reasonably detects all of the possible appliances yet, mainly because of non efficient usage of information. The process of available algorithms is shown in figures 1.1 and 1.2, about the first three steps which are the same in our proposed algorithm some explanation is given in next chapter, but the rest of the steps are completely different in our algorithm.

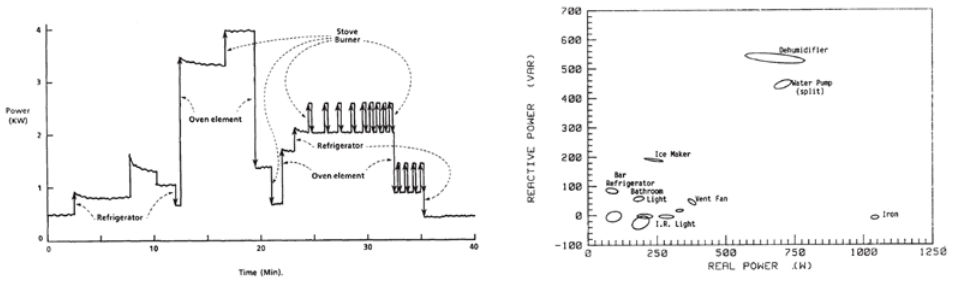

FIGURE 1.1: Total power consumption example and the related cluster of the power jumps [1]

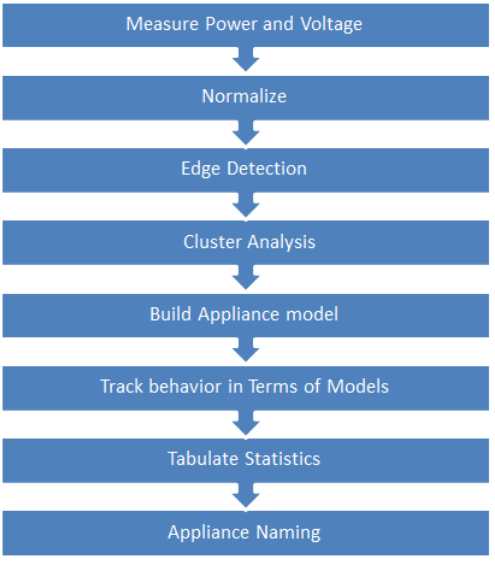

FIGURE 1.2: NIALM algorithm [1]

Viterbi is an algorithm which obtains the maximum likelihood estimation of the state change. The aim of these thesis is to propose a new approach to this problem which use available information efficiently and find the optimum solution. The Viterbi algorithm have a high potential to apply different kind of information like power, time and transition probability into its calculation.

In the next chapter, general view over the input data and appliance signatures which are used in our algorithm, is given. Explanation of the Viterbi algorithm

and hidden Markov model, which our model is based on, are given in chapter 3. Chapter 4 and 5 represent the proposed model and algorithm using the Viterbi algorithm for Markovian and non-Markovian appliances, whose difference will be explained in chapter 3. Conclusion and future work are given in chapter 6.

Chapter 2

Model input

2.1 Appliance signature

Each appliance has a unique feature in power consumption behaviour which makes it different from the others. Generally, an appliance signature can be defined as a measurable parameter of the total load that gives information about the nature or operating state of an individual appliance in the load, which also referred to as fingerprint[3]. Three classes of appliance models are defined: on or off state, Finite State Machine (FSM), continuously variable, see Figure 2.1.

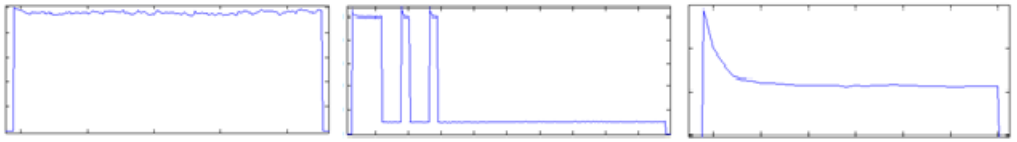

FIGURE 2.1: Appliance's model, a.on or off state, b.finite state, c.continuously variable.

FSM is an acceptable model for most household appliances. It is possible to consider on and off state and continuously variable states as an special form of FSM. As is shown in Figure 2.2, states are related to each other by allowed state transitions. States are defined based on appliance signature power levels. In this example, is assumed that the transitions are possible from first power level to the second, second to the third and from the last to the first level. In general, FSM allows for an arbitrary set of discrete states and state transitions. The p-vector signatures and their state transitions behaviour can be simply shown by FSM diagram.

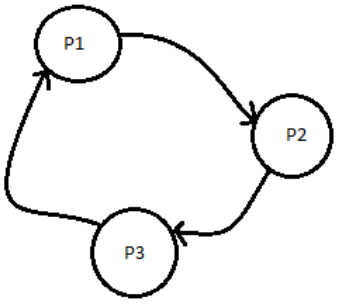

FIGURE 2.2: FSM correspond to the appliance with three different power levels.

A partial signature taxonomy is shown in Figure 2.3. The top level breakdown is between Intrusive and Non-intrusive signatures.

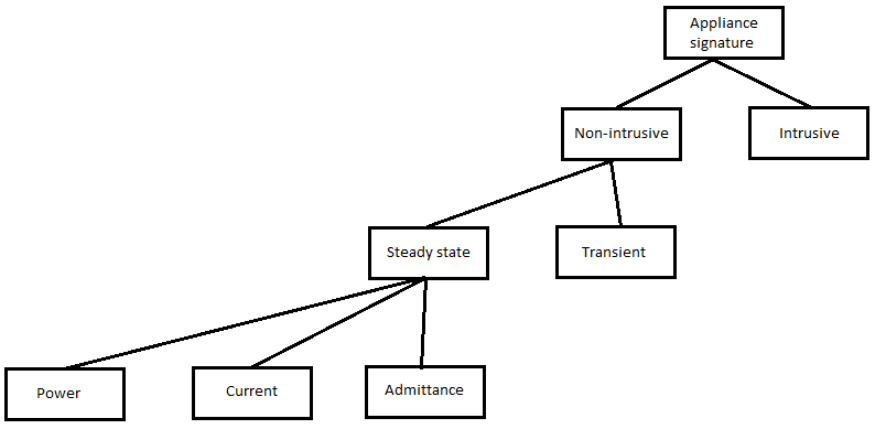

FIGURE 2.3: Partial signature taxonomy [1]

We are interested in non-intrusive signature. A non-intrusive approach is analysis of the total load operation, measuring the central load. Within the non-intrusive signatures there are two approaches to using the information about the appliance state: steady-state and transient. Steady-state signatures are continuously present during the operation time of the load, the transient signatures referred to the short period of time during the state transition. The difference between these two type of signatures are about the time of the information extraction. The steady-state signatures derive from calculating the difference between the operating levels before and after the transition. Proposed algorithm in this thesis is based on steady-state signatures, the first reason is that the steady state signatures are easier to detect, because of a continuously presence of the operation level after the transition time. The second reason is that they are additive. The activity of steady-state signatures allows to properly analysis of the simultaneous events.

The third reason is that the turn off events has no transient signature and it means the transient signature provides less information than steady state.

The steady states signatures consists of change in real and reactive power.

2.2 Real and reactive power

Considering that the standard household appliances are AC loads, the active power which is shown by P and the reactive power which is shown by Q measurements are needed. The active power consumed by the resistive load and the reactive power consumed by the inductive and capacitive loads. The complex power S is calculated by adding the vector P and Q and it indicates the power consumed by the resistive and reactive load. The electrical devices show different characteristics in the connection period based on the nature of the loads in their structure. Three groups can be defined based on these facts: purely resistive loads which can be shown by $P = Scos\Phi$, purely reactive loads which can be shown by $Q = Ssin\Phi$, and the combination of these two, see Figure 2.4. The angle Φ specify the amount of power which transferred to the active and the reactive power.

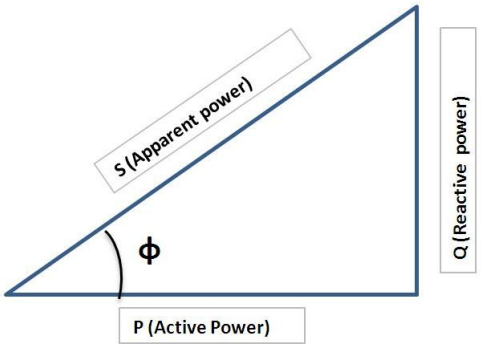

FIGURE 2.4: Power

The voltage and current are measured by 1 Hz sampling rate, average power and RMS voltage are calculated from this data. Choosing the averaging period affects the number of reported events which occur simultaneously. If the data collects slower than the 1Hz sampling rate, events which are separated by a couple of seconds will be reported as a simultaneous event. The reason that the system measure both reactive power and real power is to distinguish two appliances with the same total power draw by differences in their complex impedance. Due to the fluctuation of V, admittance is preferable to power and current as a signature, because it is more stable than the current and power signatures. A linear device

has a voltage-independent property, which means that they are additive when are wired in parallel. The load admittance, Y(t), can be calculated from the measured power P(t), and RMS voltage, V(t) as:

$$Y(t) = \frac{P(t)}{V(t)^2}. \tag{2.1}$$

Admittance is unfamiliar and we prefer to deal with admittance in the guise of normalized power[1]:

$$P_{Norm}(t) = 120^2 Y(t) = (\frac{120}{V(t)})^2 P(t). \tag{2.2}$$

This normalized power is an input of the event detector.

2.3 Events

A step change in power or the transition of an appliance's operating state to another state is labeled as an event[3]. The normalized power is an input to an edge detection algorithm which extracts the time and size of the events. Signal processing techniques, are used to find the times at which a signal changes rapidly. The overall normalised power is the sum of the normalised power of the active devices. If power consumption jump ΔP occurs for one appliance, the overall power would change by the same amount ΔP. Therefore the device characteristic features can be extracted by observing the variations in the overall normalised power. The key challenges are in coupling the variations of the normalised power to the variation of the status of a device. The event detector removes the power changes due to the transient states and also the power changes below certain threshold, assuming that these power changes are related to the noise, this is shown in Figure 2.5.

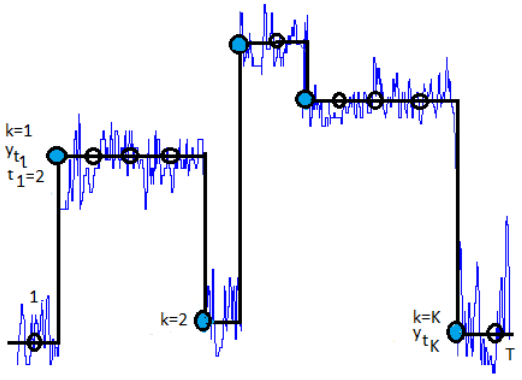

FIGURE 2.5: Observed total power

Sequence of sample times is shown by the black circles, **y**: sequence of powers at event time, **t**: sequence of event times, **k**: sequence of event numbers which are denoted by the filled in blue circles

Chapter 3

Markov Model and the Viterbi Algorithm

3.1 Markov model

In stochastic and probability theory, the Markov property refers to a memory-less property of a stochastic process. The exponential distribution as a continuous probability distribution and the geometric distribution as a discrete probability distribution are the only memory-less probability distributions. This memory-less property as a basic assumption of the Markov property indicates that the properties of random variables relevant to the future, only depend on information related to current time and not on information from the past. For example if T is considered as a waiting time for an event to occur, and τ is assumed to be some initial period of time, an exponentially distributed random variable T, see Figure 3.1, follows the rule[4]:

$$\Pr\left(T > \tau + \delta \mid T > \tau\right) = \Pr(T > \delta), \qquad \forall \tau, \delta \geq 0. \tag{3.1}$$

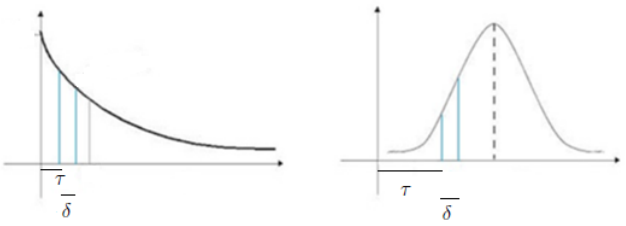

FIGURE 3.1: An exponentially distributed random variable T which describes the Markovian phenomena vs a Gaussian distributed random variable T which describes the Non-Markovian phenomena

This shows that the distribution of T given the non-occurrence of the event during the first τ seconds, is equal to the original unconditional distribution. Thus if an event has not occurred after τ seconds, the conditional probability that occurrence will take at least δ more seconds is equal to the unconditional probability of observing the event more than δ seconds relative to the initial time.

3.1.1 Markov chain

The Markov model is a stochastic model which is based on the Markov property. The first-order Markov model refers to the model in which the states at discrete time k depend only on the previous states at time $k-1$ in a non-deterministic way. If the states at time k depend on states at times $k-1$, $k-2$, ..., $k-p$, the model is called a p-order Markov model. The Markov model usually refers to the first-order Markov model.

$$Pr(s_k \mid s_{k-1}, s_{k-2}, \ldots, s_0) = Pr(s_k \mid s_{k-1}) \tag{3.2}$$

The joint probability using the Markov assumption can be defined as:

$$Pr(s_0, \ldots, s_K) = Pr(s_0).\prod_{k=1}^{K} Pr(s_k \mid s_{k-1}). \tag{3.3}$$

An example of a simple Markov model is the Markov chain model. States of Markov chains are modeled with random variables which change through time. The distribution of these variables only depends on the distribution of the previous states. The initial state s_0 and transition from each state to the other states with certain probabilities should be defined.

3.1.2 Hidden Markov Model (HMM)

Compared to the Markov chain model which has states with known random variables that change through time, in HMM these random variables are not known, which means that the states are hidden. HMM is defined where there are some observations which are related to the states, but are not sufficient to precisely determine the states, the relation between states and the observations is shown in Figure 3.2. The random variable s_k is the hidden state at time k. The random variable o_k is the observation at time k. Conditional dependencies are shown by means of arrows.

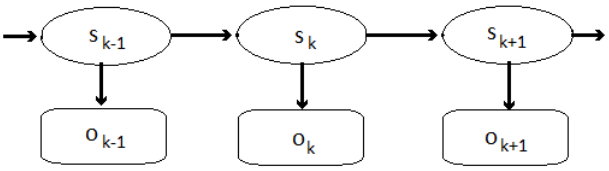

FIGURE 3.2: General architecture of the HMM

As is shown in this diagram, the conditional probability distribution of the hidden variable s_k at time k, given the values of the hidden variables of all times, $\mathbf{s}$, only depends on the value of the hidden variable s_{k-1}. The values at time s_{k-2} and before have no influence. Similarly, the value of the observed variable o_k only depends on the value of the hidden variable s_k.

Bayes' rules are applied when the states are hidden:

$$Pr(s_1, \ldots, s_K \mid o_1, \ldots, o_K) = \frac{Pr(o_1, \ldots, o_K \mid s_1, \ldots, s_K) Pr(s_1, \ldots, s_K)}{Pr(o_1, \ldots, o_K)}, \quad (3.4)$$

where $o_1, o_2, \ldots, o_K$ are the observations during the K discrete times, $Pr(s_1, \ldots, s_K \mid o_1, \ldots, o_K)$ is the probability of states given the observation, which is desired, $Pr(o_1, \ldots, o_K)$ is the prior probability of seeing a particular sequence of observation, and finally:

$$Pr(o_1, \ldots, o_K \mid s_1, \ldots, s_K) = \prod_{k=1}^{K} Pr(o_k \mid s_k). \quad (3.5)$$

Assuming that each state at time k has I possible states which can be shown as $\mathbf{s}_k = s_{k1}, \ldots, s_{kI}$, the observation at $\mathbf{s}_k$ is o_k:

$$Pr(o_1, \ldots, o_K \mid \mathbf{s}_1, \ldots, \mathbf{s}_K) = \prod_{k=1}^{K} Pr(o_k \mid \mathbf{s}_k). \quad (3.6)$$

Assuming that for all k, given $\mathbf{s}_k$ and o_k are independent of all $\mathbf{s}_j$ and o_j, for all $k \neq j$, $j = 1, \ldots, K$, the purpose is to find the most likely sequence of states given some observation input:

$$\arg\max Pr(\mathbf{S} \mid \mathbf{o}) = \arg\max \frac{Pr(\mathbf{o} \mid \mathbf{S}) Pr(\mathbf{S})}{Pr(\mathbf{o})}, \quad (3.7)$$

where $\mathbf{o} = o_1, o_2, \ldots, o_K$ is the sequence of observations during the K discrete times and $\mathbf{S} = \mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_K$ is the sequence of state vectors $\mathbf{s}_k$. The input $\mathbf{o}$ for a one dimensional observation will be constant, and so will $Pr(\mathbf{o})$, thus it is only

required to find:

$$\arg\max Pr(\mathbf{S} \mid \mathbf{o}) = \arg\max Pr(\mathbf{o} \mid \mathbf{S}) Pr(\mathbf{S}). \tag{3.8}$$

In the standard type of hidden Markov model which is considered here, the finite state space $\mathbf{s}_k$ of the hidden variables is discrete and this hidden state space is assumed to consist of one of I possible values. The observations can either be discrete or continuous. Discrete observations are typically generated from a categorical distribution as the most general distribution over a K-way event. The continuous observation is typically generated from a Gaussian distribution.

Two types of parameters are defined in a hidden Markov model, transition probabilities and emission probabilities. The transition probabilities are the probability that the state at time k is chosen given the hidden state at time $k-1$. This means that for each state at time k which has I possible states, there is a transition probability from state $k-1$ which also has I possible states, given by the $I \times I$ matrix of transition probabilities which is a Markov matrix. Any one transition probability can be determined once the others are known, thus there are a total of $I(I-1)$ independent transition parameters. The hidden state space is modeled as a categorical distribution, the set of transition probabilities for transitions from any given state must be in the range 0 to 1, and all must add up to 1.

There is a set of emission probabilities for each of the possible I states, which is governing the distribution of the observed variable at a particular time, given the state of the hidden variable at that time. If the observed variable is discrete with K possible values, governed by a categorical distribution, there will be $K-1$ separate parameters, for a total of $I(M-1)$ emission parameters over all the hidden states[4][5].

3.2 Viterbi Algorithm (VA)

The VA is an algorithm which finds the most likely path given a set of observations through a trellis. The trellis in this case represents a graph of a finite set of states from a FSM. Each node in this graph represents a state and each edge a possible transition between two states at consecutive discrete time intervals. The trellis nodes are ordered into vertical discrete time slices and each node at each time connected to at least one node at an earlier and at least one node at a later time. The earliest and latest times in the trellis have only one node.

An example of a trellis is shown in Figure 3.3, and the FSM that produced this trellis is shown in Figure 3.4. Often this FSM model is referred to as a Markov model.

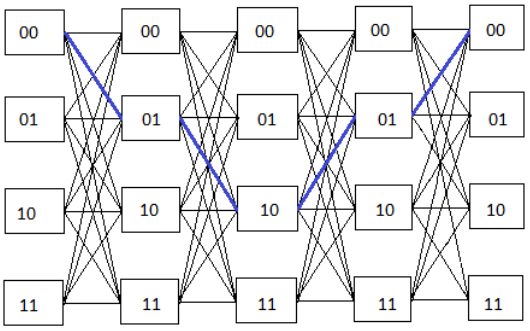

FIGURE 3.3: Trellis

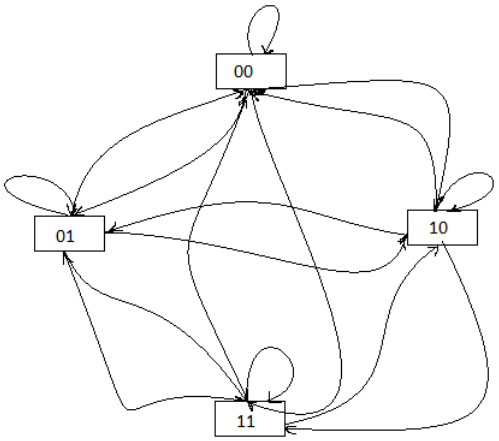

FIGURE 3.4: An example of FSM

For each of the possible transitions within a given FSM there is a corresponding output produced by the FSM. The outputs of the FSM are viewed by the VA as a set of observations.

If the structure of the FSM is known, the case is similar to the Markov Model, it means that the states have a known random variable. Another type of FSM is the Hidden Markov Model. As the name suggests the actual FSM is hidden from the VA and has to be viewed through the observations produced by the HMM. In this case the trellis's states and transitions are estimates of the underlying HMM, where only estimates of the true state of the system can be produced.

The VA presents the maximum likelihood estimation $\hat{\mathbf{s}}$ of the state metric $\mathbf{S}$, given the metric of observations $\mathbf{o}$.

$$\hat{\mathbf{s}} = \arg\max Pr(\mathbf{S} \mid \mathbf{o}) = \arg\max Pr(\mathbf{S})Pr(\mathbf{o} \mid \mathbf{S}). \tag{3.9}$$

The algorithm calculates the maximum probability of observing a state after each transition step considering the previous state. In the last step, the state with maximum probability is selected and traced back to recover the most likely transition sequence.

The Viterbi algorithm's joint probability evaluation in a HMM:

$$Pr(\mathbf{S}, \mathbf{o}) = Pr(\mathbf{S})Pr(\mathbf{o} \mid \mathbf{S}) = Pr(\mathbf{s}_0 \mid \boldsymbol{\pi}) \cdot \prod_{k=1}^{K} Pr(\mathbf{s}_k \mid \mathbf{s}_{k-1}) \cdot \prod_{k=1}^{K} Pr(o_k \mid \mathbf{s}_k). \tag{3.10}$$

k - The discrete time index.

I - State space is assumed to consist of I possible values.

$\mathbf{s}_k$ - The state vector $\mathbf{s}_k = [1\ 2 \dots I]$ of the FSM at time k.

o_k - The observation at time k.

$\mathbf{s}$ - The survivor path which terminates at time K, in the s_kth state of the FSM. It consists of an ordered list of s_k's visited by this path from time $k = 0$ to time k.

K - Truncation length of the VA. It is the time when a decision has to be made by the VA.

$\boldsymbol{\pi}$ - Initial state vector for the ith state at $k = 0$. Defined as the probability that the ith state is the starting state, $Pr(s_0)$.

In either type of model, MM or HMM, the VA uses a set of metrics associated with the observation and the transitions within the FSM. These metrics are used to calculate the cost of the various paths through the trellis, and are used by the VA to decide which path is the most likely path to have been followed, given the set of observation.

$\mathbf{A}$ - The transition metric. The transition vector $\mathbf{a}_k$, defined as the probability that given that state s_{k-1} occurs at time $k-1$, the state s_k will occur at time k, $a_{s_{k-1}s_k} = Pr(s_k \mid s_{k-1})$.

E - The observation metric. The observation vector $\mathbf{e}_k$, defined as the probability that the observation o_k would occur at time k, given that we are in the state s_k at time k, $e_k = Pr(o_k \mid s_k)$.

State - The state's survivor path metric. This is defined as the product of the vector $\boldsymbol{\pi}$ and the metrics (**A** and **E**) for each transition in the s_kth survivor path, from time $k = 0$ to time K.

The most likely transition at each state s_k at time k, is chosen. If two or more transitions are found to be maximum, then one of the transitions is chosen randomly as the most likely transition. These states are added to the survivor path of the states s_o to s_{k-1} at time k, and rest of the states are discarded. This then becomes the survivor path of the state s_k at time k. The same operation is carried out until reach time $k = K$, at the decision point K the state which contains the maximum probability is chosen. The VA then outputs this estimated survivor path $\hat{\mathbf{s}}$, along with it's survivor metric **State**[5][6].

Chapter 4

Viterbi detector applied to power disaggregation for Markovian appliance signatures.

4.1 Problem Description

Given a discrete sequence of powers after event detection, $\mathbf{y} = y_{t_1}, y_{t_2}, \ldots, y_{t_K}$, which is shown here as $\mathbf{y} = y_1, y_2, \ldots, y_K$ for simplicity, and discrete sequence of time $\mathbf{t} = t_1, t_2, \ldots, t_K$, which is the sequence of detected event times, determine the sequence of appliances states, $\mathbf{s} = s_1, s_2, \ldots, s_K$. k is one of the K discrete times, based on the detected event times, $\mathbf{k} = 1, \ldots, K$. The sequences $\mathbf{y}$, $\mathbf{t}$ and $\mathbf{k}$ are shown in figure 2.5.

4.2 The Viterbi model corresponds to the proposed algorithm

Assuming that the discrete time index k is based on the detected event times, as was mentioned in the previous chapter, the truncation length of the VA is K. Each appliance n, with $n = 1, \ldots, N$, might have a different number of power levels. The sequence of $\mathbf{L}$ is defined as $\mathbf{L} = L^{(1)}, \ldots, L^{(N)}$, where each $L^{(n)}$ represents the total number of power levels for appliance n. The appliance n at time k is at state $s_k^{(n)}$, which means that corresponds to the $L^{(n)}$ at time k, for appliance

n, $s_k^{(n)} = 1, \ldots, L^{(n)}$. The total state of the appliances at time k is defined as $\mathbf{s}_k = [s_k^{(1)}\ s_k^{(2)} \ldots s_k^{(N)}]$.

The number of states of this Markov model is $I = \prod_{n=1}^{N} L^{(n)}$, where $s_k = 1, \ldots, I$, is the sequence of states numbers at each k. For simplicity, it is assumed that all the appliances have the same number of power levels L, therefore $I = L^N$.

In the next two sections, two algorithms are proposed for the power only detector and the Markovian appliance detector. The observation corresponding to the first algorithm is the discrete sequence of powers, $\mathbf{y}$, and for the second algorithm, two sequences of observations are given, these are $\mathbf{y}$ and $\mathbf{t}$.

4.3 Power only algorithm, no time information

Appliance signatures can be defined by the sequence of the power levels. There are several ways to distinguish the appliances with different power levels, but there are many appliance signatures with the same power level sequence or some similarity in their power level sequences.

As was mentioned before, distinguish between the appliances with some similarity in their power level sequences, is a challenge for available algorithms. Here using the Viterbi Algorithm and proposed model, one solution to this problem is proposed. If the power signatures have any difference in their power level, no matter how similar the rest of the sequence is, it is possible to distinguish them with a high level of accuracy using the given algorithm in this section.

If different signatures have exactly the same power level sequences, it is obvious that more information is needed to detect them correctly. In the next section, the time information is applied into the algorithm to improve it.

4.3.1 Proposed appliance model

For each appliance n, $n = 1, \ldots, N$, we have defined $\theta_y^{(n)} = \{\mathbf{\Phi}_y^{(n)}, \mathbf{A}^{(n)}, \boldsymbol{\pi}^{(n)}\}$[7], respectively corresponding to the probability that power observation was generated by an appliance state, the transition probabilities between states and the probability of an appliance's initial state. To define the emission density it is assumed that each appliance has a Gaussian power distribution for each power level, which is defined by $(\mu_y)_{s_k}^{(n)}$ and $(\sigma_y)_{s_k}^{(n)}$, where $s_k^{(n)}$ is the index of the appliance's power level, $s_k^{(n)} = 1, \ldots, L^{(N)}$. An appliance's signature is shown in Figure 4.1.

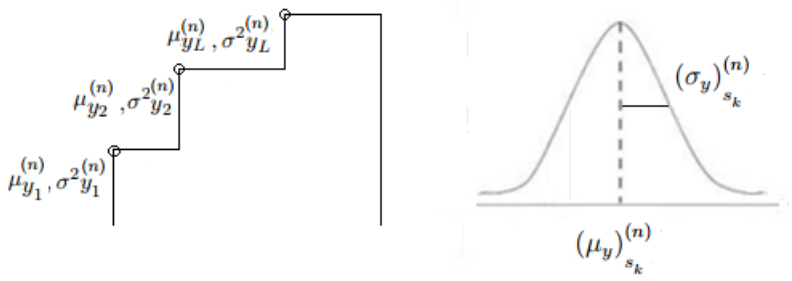

FIGURE 4.1: Appliance n signature and power level $s_k^{(n)}$ distribution.

As is shown in this figure the appliance n has 3 power levels ($L = 3$), and each power level has its own Gaussian power distribution. $s_k^{(n)} = 1, 2, 3$

Each appliance has a transition matrix which shows the probability of state change between different power levels of that appliance. This is shown by $\mathbf{A}^{(n)}$, where $a_{s_{k-1}^{(n)} s_k^{(n)}}$ is the probability of transition from the state $s_{k-1}^{(n)}$ to the state $s_k^{(n)}$, for appliance n.

The probability of an appliance's initial state $\boldsymbol{\pi}^{(n)}$, represents that the probability of that the appliance n being at level $s_0^{(n)}$ at $k = 0$, is equal to $\pi_{s_0}^{(n)}$, $s_0^{(n)} = 1, \ldots, L^{(n)}$.

In this section, as a first approach, the power information is used as an input for the VA. The VA will be applied to the power and time input simultaneously in the next section.

4.3.2 Proposed algorithm, power only detector

The probability of the appliance's starting state s_0 is defined as:

$$Pr(s_0) = \pi_{s_0}\ , s_0 = 1, \ldots, I. \tag{4.1}$$

As was mentioned before, s_k is a decimal number corresponding to the total state of appliances at time k, the total state s_k can also be represented by the vector $\mathbf{s}_k$, where $\mathbf{s}_k = [s_k^{(1)}\ s_k^{(2)} \ldots s_k^{(N)}]$, each $s_k^{(n)}$ indicates the state of appliance n at time k, where $s_k^{(n)} = 1, \ldots, L^{(n)}$. If it is assumed that $L^{(N)}$ is equal for all the appliances and $L^{(N)} = L$, this vector $\mathbf{s}_k$, represents the number s_k in L-base.

The probability that state $\mathbf{s}_0 = [s_0^{(1)}\ s_0^{(2)} \ldots s_0^{(N)}]$, is equal to:

$$\pi_{s_0} = \prod_{n=1}^{N} \pi_{s_0}^{(n)}. \tag{4.2}$$

$$\boldsymbol{\pi} = \pi_1, \ldots, \pi_I \tag{4.3}$$

The transition probabilities from states $\mathbf{s}_{k-1}^{(n)}$ to state $\mathbf{s}_k^{(n)}$ are given by the transition matrix $A^{(n)}$:

$$Pr(s_k \mid s_{k-1}) = \prod_{n=1}^{N} Pr(s_k^{(n)} \mid s_{k-1}^{(n)}). \tag{4.4}$$

$$Pr(s_k^{(n)} \mid s_{k-1}^{(n)}) = \left(a_{s_{k-1}^{(n)} s_k^{(n)}}\right). \tag{4.5}$$

Notation: $(X)_{mn}$ refers to the matrix X's element which is on row m and in column n.

Assume that given its state, each appliance has a Gaussian distribution in power:

$$y_k \mid s_k, \boldsymbol{\phi}_y \sim N(\sum_{n=1}^{N} (\mu_y)_{s_k}^{(n)}), \sum_{n=1}^{N} (\sigma_y)_{s_k}^{(n)}). \tag{4.6}$$

According to the VA's joint probability evaluation in an HMM:

$$\begin{aligned} Pr(\mathbf{S} \mid \mathbf{y}, \boldsymbol{\theta}_y) &= Pr(\mathbf{y} \mid \mathbf{S}) Pr(\mathbf{S} \mid \boldsymbol{\theta}_y) \\ &= Pr(\mathbf{s}_0 \mid \boldsymbol{\pi}) \prod_{k=1}^{K} Pr(\mathbf{s}_k \mid \mathbf{s}_{k-1}) \cdot \prod_{k=1}^{K} Pr(y_k \mid \mathbf{s}_k). \end{aligned} \tag{4.7}$$

4.3.2.1 Simulation1 assumptions

To verify the accuracy of this algorithm, 6 appliances are defined based on their sequence of power levels, which are shown in Figure 4.2.

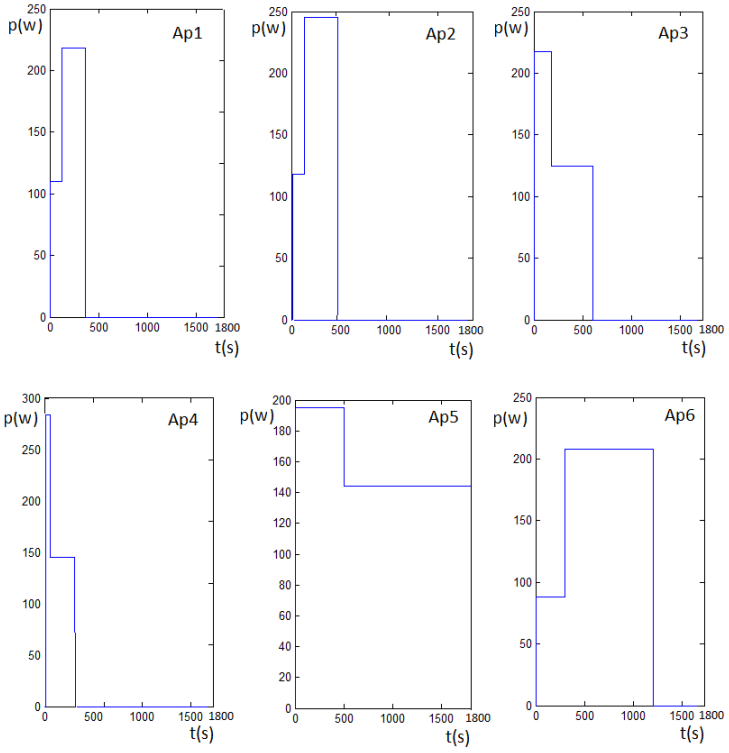

FIGURE 4.2: Appliance signatures, real power

For the sake of simplicity, the number of power levels of the all appliances is assumed to be equal to three. These three levels are defined as off, on1 and on2. Each power level has a Gaussian distribution. As is shown in Table 4.1, Appliances 1 to 5 are considered to be partly identical and partly different in terms of the sequence of power levels. The reason behind this, is to show whether the proposed algorithm is able to detect the appliance correctly in the presence of appliances which have similar power levels, by using the VA to find the most likely path. Appliances 1 and 6 are considered to be identical in the sequence of power levels, to show that the proposed algorithm is not able to distinguish between them by considering only the power information. In the next section, the time information will be applied to improve the detection accuracy.

n	μ_1, σ_1^2	μ_2, σ_2^2	μ_3, σ_3^2
1	$0, 10$	$100, 10$	$200, 10$
2	$0, 10$	$100, 10$	$250, 10$
3	$0, 10$	$200, 10$	$150, 10$
4	$0, 10$	$300, 10$	$150, 10$
5	$0, 10$	$200, 10$	$100, 10$
6	$0, 10$	$100, 10$	$200, 10$

TABLE 4.1: Emission density $\mathbf{\Phi}_y^{(n)}$, simulation1

$N = 6, L = 3$, appliances 1 and 6 with the same power sequences

The initial state and transition matrices for all 6 appliances are considered to be the same, see in this connection Figure 4.3 and Tables 4.2, 4.3. It is assumed that the transition from off to on2, on2 to on1 and on1 to off is zero, which means that all the appliances here have a fixed order of operation. This assumption is just for the sake of simplicity. For non-fixed order operation, the algorithm will be the same and just the transition matrix must be modified .

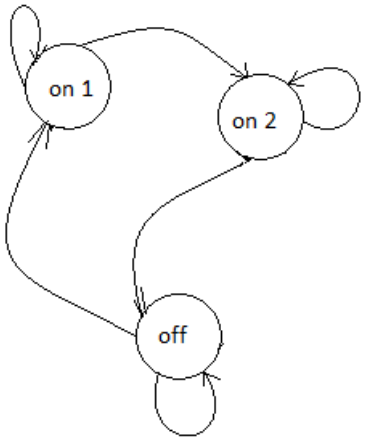

FIGURE 4.3: State transition model, simulation1

l	1 :off	2 :on1	3 :on2
1 :off	2/3	1/3	0
2 :on1	0	2/3	1/3
3 :on2	1/3	0	2/3

TABLE 4.2: Transition matrix $\mathbf{A}^{(n)}$, simulation1

$s_{k-1}^{(n)} = 1 : off, 2 : on1, 3 : on2$, $s_k^{(n)} = 1 : off, 2 : on1, 3 : on2$, $n = 1, \ldots, 6$

For initial state, it is assumed that there is no possibility to start from state on2 for none of the appliances. The probability that each appliance starts from off is twice the probability of start from state on1, there is no certain reason for these assumptions, the initial state matrix can be filled in by any other probabilities based on given data.

n	1 :off	2 :on1	3 :on2
1	2/3	1/3	0
2	2/3	1/3	0
3	2/3	1/3	0
4	2/3	1/3	0
5	2/3	1/3	0
6	2/3	1/3	0

TABLE 4.3: Initial state $\boldsymbol{\pi}^{(n)}$, simulation1

$s_0^{(n)} = 1 : off, 2 : on1, 3 : on2$, the probabilities are chosen the same for all appliances in the sake of simplicity.

One of the random results from given assumptions is shown in Figure 4.4, this is the input of power only detector, the detection results are given in the next part.

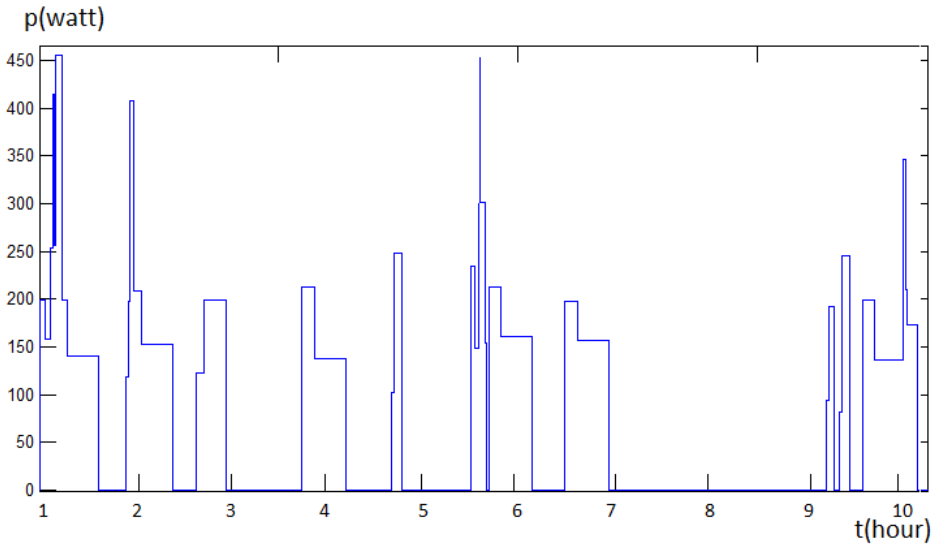

FIGURE 4.4: Observation, simulation1

4.3.2.2 Simulation1 results

As is shown below, if the appliances have any difference in their sequence of powers, they can be perfectly detected, as is shown in Table 4.4.

k	$\hat{\mathbf{S}}_k$						$\mathbf{S}_k$					
	$s_k^{(1)}$	$s_k^{(2)}$	$s_k^{(3)}$	$s_k^{(4)}$	$s_k^{(5)}$	$s_k^{(6)}$	$s_k^{(1)}$	$s_k^{(2)}$	$s_k^{(3)}$	$s_k^{(4)}$	$s_k^{(5)}$	$s_k^{(6)}$
0	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	2	1	1	1	1	1	2	1	1	1
2	1	1	3	1	1	1	1	1	3	1	1	1
3	1	2	3	1	1	1	1	2	3	1	1	1
4	1	3	3	1	1	1	1	3	3	1	1	1
5	1	3	1	1	1	1	1	3	1	1	1	1
6	1	3	1	1	2	1	1	3	1	1	2	1
7	1	1	1	1	2	1	1	1	1	1	2	1
8	1	1	1	1	3	1	1	1	1	1	3	1
9	1	1	1	1	1	1	1	1	1	1	1	1

TABLE 4.4: Result, the estimated states are equal to the appliance states, simulation1

As was mentioned, appliances 1 and 6 have exactly the same sequence of power levels, this will lead to an error on state $k = 17$ to 19, see Table 4.5, where appliance 6 is on but the detector shows that appliance 1 is on, 1 is randomly chosen over 6. In this kind of situation we need to use some extra information about the time duration of the signatures or some other features of the appliances to distinguish them.

k	$\hat{\mathbf{S}}_k$						$\mathbf{s}_k$					
	$s_k^{(1)}$	$s_k^{(2)}$	$s_k^{(3)}$	$s_k^{(4)}$	$s_k^{(5)}$	$s_k^{(6)}$	$s_k^{(1)}$	$s_k^{(2)}$	$s_k^{(3)}$	$s_k^{(4)}$	$s_k^{(5)}$	$s_k^{(6)}$
10	1	1	1	1	1	1	1	1	1	1	1	1
11	2	1	1	1	1	1	2	1	1	1	1	1
12	3	1	1	1	1	1	3	1	1	1	1	1
13	3	1	1	1	2	1	3	1	1	1	2	1
14	1	1	1	1	2	1	1	1	1	1	2	1
15	1	1	1	1	3	1	1	1	1	1	3	1
16	1	1	1	1	1	1	1	1	1	1	1	1
17	2	1	1	1	1	1	1	1	1	1	1	2
18	3	1	1	1	1	1	1	1	1	1	1	3
19	3	1	2	1	1	1	1	1	2	1	1	3
20	1	1	2	1	1	1	1	1	2	1	1	1
21	1	1	3	1	1	1	1	1	3	1	1	1
22	1	1	1	1	1	1	1	1	1	1	1	1

TABLE 4.5: Result, the estimated states are not equal to the appliance states, simulation1

at k : $17, 18, 19$, where the appliances 1 and 6 have the same power sequences but different time sequences

4.4 Markovian appliance

An algorithm will be discussed in this section, to distinguish between two appliances with the same power signatures but different time features. This algorithm

represents an improvement to the previous algorithm but only for Markovian appliances.

4.4.1 Proposed Markovian appliance model

In this part, the information of the average time for power levels of each appliance is used to improve the previous algorithm, see Figure 4.5. The appliance's features are shown by $\theta_{y,t}^{(n)} = \{\boldsymbol{\pi}^{(n)}, \boldsymbol{\Phi}_y^{(n)}, \boldsymbol{\Phi}_t^{(n)}\}$, respectively corresponding to the probability of an appliance's initial state, the probability that a power observation was generated by an appliance state, and the average time for the power levels of each appliance. By knowing that the appliances are Markovian, the given information at each step is from the current and previous states. As was discussed before, the exponential distribution is the only continuous distribution which has the Markov property. To define the time emission density, it is assumed that each appliance has an exponential time distribution for each signature level, feature of the exponential time distribution for each power level of each appliance is shown by $(\mu_t)_{s_k}^{(n)}$, where $s_k^{(n)}$ is the index of the appliance's signature level. All the distributions are independent

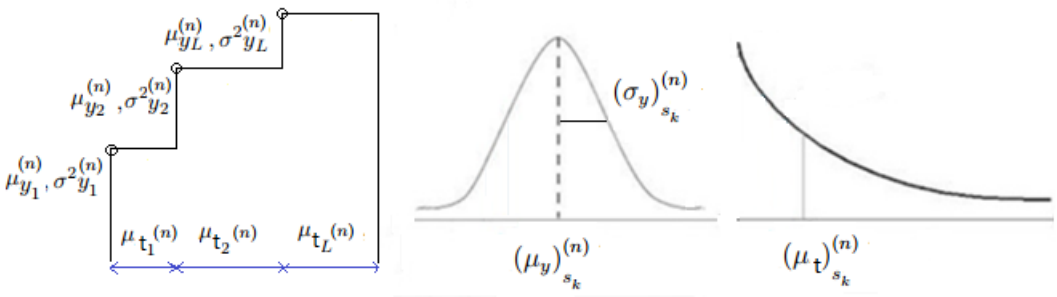

FIGURE 4.5: Markovian appliance's signature

4.4.2 Proposed algorithm, an exponential detector

In this part by applying the time information into the Viterbi algorithm, we expect some improvement in detection results. Given the sequences of $\mathbf{y}, \mathbf{t}$, the sequence of $\hat{\mathbf{s}}$ could be estimated as:

$$\begin{aligned}\hat{\mathbf{s}} = \arg\max Pr(\mathbf{S} \mid \mathbf{y}, \mathbf{t}) &= \arg\max \frac{Pr(\mathbf{y} \mid \mathbf{S}, \mathbf{t}) Pr(\mathbf{S}, \mathbf{t})}{Pr(\mathbf{y}, \mathbf{t})} \\ &= \arg\max Pr(\mathbf{y} \mid \mathbf{S}) Pr(\mathbf{S}, \mathbf{t}),\end{aligned} \tag{4.8}$$

it is because the $Pr(\mathbf{y}, \mathbf{t})$ is constant, and the probability of that the observed power belongs to a certain state depends only on the states and not on the observation time. Using the Markov property:

$$Pr(\mathbf{S}, \mathbf{t}) = Pr(\mathbf{s}_0, t_0) \prod_{k=1}^{K} Pr(\mathbf{s}_k, t_k \mid \mathbf{s}_{k-1}, t_{k-1}). \tag{4.9}$$

In continuous time, it is assumed that at each event only one device state change is possible. $s_k^\star$ and $s_{k-1}^\star$ are the pair of states with maximum one appliance at different state, $n^\star$ is the appliance whose state changes in state transition from $s_{k-1}^\star$ to $s_k^\star$.

According to the VA's joint probability evaluation in an HMM:

$$\begin{aligned} Pr(\mathbf{S}^\star \mid \mathbf{y}, \mathbf{t}, \boldsymbol{\theta}_{y,t}) &= Pr(\mathbf{y} \mid \mathbf{S}^\star, \boldsymbol{\theta}_{y,t}) Pr(\mathbf{S}^\star, \mathbf{t}) \\ &= Pr(\mathbf{s}_0^\star, t_0) \textstyle\prod_{k=1}^{K} Pr(\mathbf{s}_k^\star, t_k \mid \mathbf{s}_{k-1}^\star, t_{k-1}) \cdot \prod_{k=1}^{K} Pr(y_k \mid \mathbf{s}_k^\star). \end{aligned} \tag{4.10}$$

The difference between this equation and equation 4.7 is in transition matrix definition, which is now based on time and state, and not only the state. To find this transition probability corresponds to each state transition we need some definition: The probability density that event k takes place time δ after event $k-1$, given that the state vector after event $k-1$ is $\mathbf{s}_{k-1}^*$, has shown by $f(\delta \mid \mathbf{s}_{k-1}^*)$, and the probability that the state vector after event k is $\mathbf{s}_k^*$, given that the sate vector after event $k-1$ is $\mathbf{s}_{k-1}^*$ and that the time between events $k-1$ and k is δ, has shown by $Pr[\mathbf{s}_k^* \mid \mathbf{s}_{k-1}^*, \delta]$.
When the state vector is $\mathbf{s}_{k-1}^*$, device n is in state ${\mathbf{s}_{k-1}^*}^{(n)}$ (the n-th component of the state vector).
Let the internal clock of device n be $\tau_{k-1}^{(n)}$. This clocks measures how long device n has been in state $s_{k-1}^{(n)}$. For the device that changed state at event $k-1$ this clock equals zero, for the other devices it is positive.
Let $G_n(x, s)$ denote the CDF of the total lifetime of device n in state s, i.e. $Pr[(\text{total lifetime in state s}) < x]$. And define $F_n(x, s)$ as the CDF of the remaining lifetime of device n in state s after the event i.e:

$$\begin{aligned} F_n(x, s) &= Pr[(\text{remaining lifetime in state s after event k-1}) < x] \\ &= Pr[\text{total lifetime in state s} < x + \tau \mid \text{total lifetime} > \tau] \\ &= \tfrac{Pr[\tau < \text{lifetime} < \tau + x]}{Pr[\text{lifetime} > \tau]} = \tfrac{(G_n(x+\tau, s) - G_n(\tau, s))}{(1 - G_n(\tau, s))}. \end{aligned} \tag{4.11}$$

With $x = \delta$, $s = \mathbf{s}_{k-1}^{(n)}$ and $\tau = \boldsymbol{\tau}_{k-1}^{(n)}$.
So we have N random numbers, denoted T_n, $1 <= n < N$, each one distributed according to its own CDF, F_n. One of these random numbers is the smallest. It means that:

$$Pr[s_k^* \mid s_{k-1}^*, \delta] = Pr[\arg min(T_1, ..T_N) = n^* \mid s_{k-1}^*, \delta]. \tag{4.12}$$

The CDF of the smallest random number due to the independency of the appliances is:

$$F_{min}(x) = 1 - \prod_{n=1}^{N}(1 - F_n(x)). \tag{4.13}$$

Taking the derivative with respect to x gives the PDF:

$$\begin{array}{rl} f_{min}(x) & = \sum_{m=1}^{N} f_m(x) \prod_{n=1, n\neq m}^{N}(1 - F_n(x)) \\ & = \sum_{m=1}^{N} \frac{f_m(x)}{1-F_m(x)} \prod_{n=1}^{N}(1 - F_n(x)). \end{array} \tag{4.14}$$

From this we see that the PDF $f_{min}(\delta) = f(\delta \mid s_{k-1}^*)$.
The contribution from device n equals:

$$\frac{f_n(x)}{1 - F_n(x)} \prod_{m=1}^{N}(1 - F_m(x)). \tag{4.15}$$

If $n^\star$ is the index of an appliance which achieves the minimum, it contributes a fraction:

$$\frac{f_{n^*}(x)/1 - F_{n^*}(x)}{\sum_{m=1}^{N} f_m(x)/1 - F_m(x)} = Pr[\arg min(T_1, ..., T_N) = n^* \mid s_{k-1}^*, \delta], \tag{4.16}$$

of all the total PDF. This fraction is precisely the probability that device $n^\star$ is the appliance causing the event.
Taking the product of the two, we get the combined probability / PDF that the next state vector is $\mathbf{s}_k^*$ and that event k takes place time δ after event $k-1$:

$$\begin{array}{rl} Pr[s_k^*, t_k - t_{k-1}\text{in}[\delta, \delta + dt) \mid s_{k-1}^*] & = f_{min}(\delta) \cdot dt \cdot \frac{f_{n^*}(\delta)/(1-F_{n^*}(\delta))}{\sum_{m=1}^{N} f_m(\delta)/(1-F_m(\delta))} \\ & = \frac{f_{n^*}(\delta)}{1-F_{n^*}(\delta)}(1 - F_{min}(\delta)). \end{array} \tag{4.17}$$

(Note that if G is exponential, In Equation 4.11, $F_n(x, s) = G_n(x, s)$, independent of τ.)
The Probability Density Function (PDF) of an exponential distribution is given

by:

$$g_{T_{s_k}^{(n)}}(x) = \begin{cases} \frac{1}{(\mu_t)_{s_k}^{(n)}} \exp(-x/(\mu_t)_{s_k}^{(n)}) & x \geq 0 \\ 0 & x < 0. \end{cases} \tag{4.18}$$

The Cumulative Distribution Function (CDF) of an exponential distribution is given by:

$$G_{T_{s_k}^{(n)}}(x) = Pr[T_{s_k}^{(n)} < x] = 1 - \exp(-x/(\mu_t)_{s_k}^{(n)}) \quad x \geq 0. \tag{4.19}$$

$$Pr[s_k^*, t_k - t_{k-1}\text{in}[\delta, \delta + dt) \mid s_{k-1}^*] = \frac{1}{(\mu_t)_{s_k}^{(n\star)}} \cdot exp(\frac{-\delta}{\mu_t}), \tag{4.20}$$

$$\frac{1}{\mu_t} = \frac{1}{(\mu_t)_{s_k}^{(1)}} + \ldots + \frac{1}{(\mu_t)_{s_k}^{(N)}} \tag{4.21}$$

which means that the Markovian model does not depend on the information from the time before $k-1$ as we expected, and $\tau_{k-1}^{(n)}$ measurements are not needed.

4.4.2.1 Simulation2 assumptions

To verify the accuracy of the new detector and compare it to the previous one, two sorts of data are generated. First, it is assumed that all the appliances have the Gaussian power level distributions and the exponential time distributions. Emission densities correspond to these distributions are shown in Tables 4.6 and 4.7. It is assumed that the start state is equal to one, where all the appliances are in off level. There is no transition matrix which depends only on states, for the new detector, but this transition matrix is defined for power only detector. The power only detector is also applied to the new data, to compare the result of these two detectors. The transition matrix for power only detector is the same as simulation1, see table 4.2. It is assumed that at each k, just one appliance state change occurs .

To generate the first set of data, at each state, based on the exponential time distribution of each appliance, a random waiting time on the current level is generated. The minimum time has chosen, the state changes correspond to the appliance with minimum waiting random time. The power levels are generated based on the Gaussian distributions.

n	μ_1, σ_1^2	μ_2, σ_2^2	μ_3, σ_3^2
1	$0, 1$	$100, 1$	$200, 1$
2	$0, 1$	$100, 1$	$300, 1$
3	$0, 1$	$600, 1$	$400, 1$
4	$0, 1$	$500, 1$	$100, 1$
5	$0, 1$	$200, 1$	$100, 1$
6	$0, 1$	$100, 1$	$200, 1$

TABLE 4.6: Emission density $\mathbf{\Phi}_y^{(n)}$ for appliance n, simulation2

$N = 6, L = 3$

n	μ_1	μ_2	μ_3
1	1100	400	600
2	4000	500	130
3	360	50	250
4	18000	180	420
5	10000	120	340
6	300	100	100

TABLE 4.7: Emission density $\mathbf{\Phi}_t^{(n)}$ for appliance n, simulation2

$N = 6, L = 3$

The second set of data is generated based on the almost deterministic signatures in time, which have the Gaussian distributions with a very low variances equal to 1. The power distributions are the same as the first set, but the mean times are given in Table 4.7. To generate this set of data, at the first stage, each appliance is generated based on the corresponding Gaussian distribution in power and time. Then the signatures are summed in time to produce the total power consumption. The aim of applying this data is to figure out if the new detector based on Markovian behaviour can detect the non-Markovian appliances which have the Gaussian distributions in time.

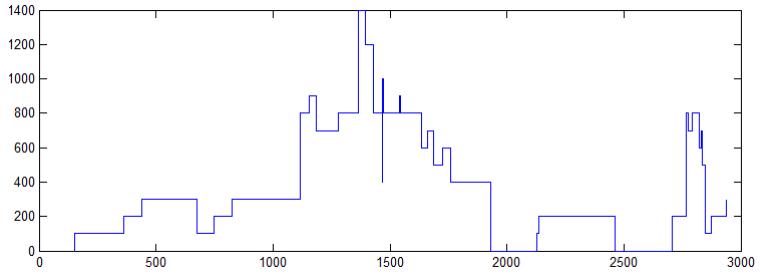

FIGURE 4.6: Exponential in time generated data

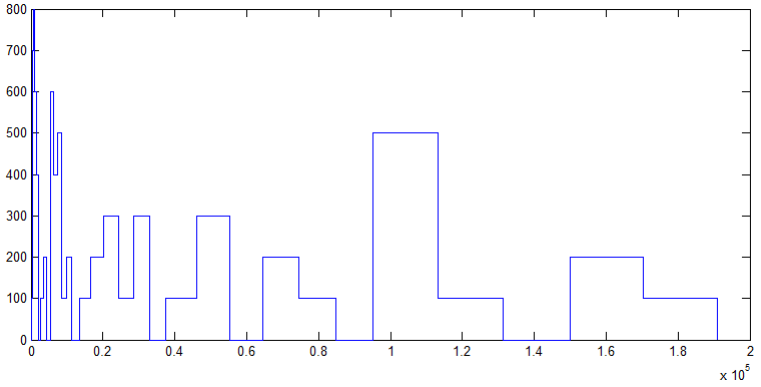

FIGURE 4.7: Deterministic in time generated data

The first set of generated data is depicted in Figure 4.6 and the second set is depicted in Figure 4.7

4.4.2.2 Simulation2 results

As is shown in Figure 4.8 the result of the detector which uses time and power information is much better than the one with only power information, where appliances have exactly the same or very close power levels, this detector can distinguish them with a good accuracy.

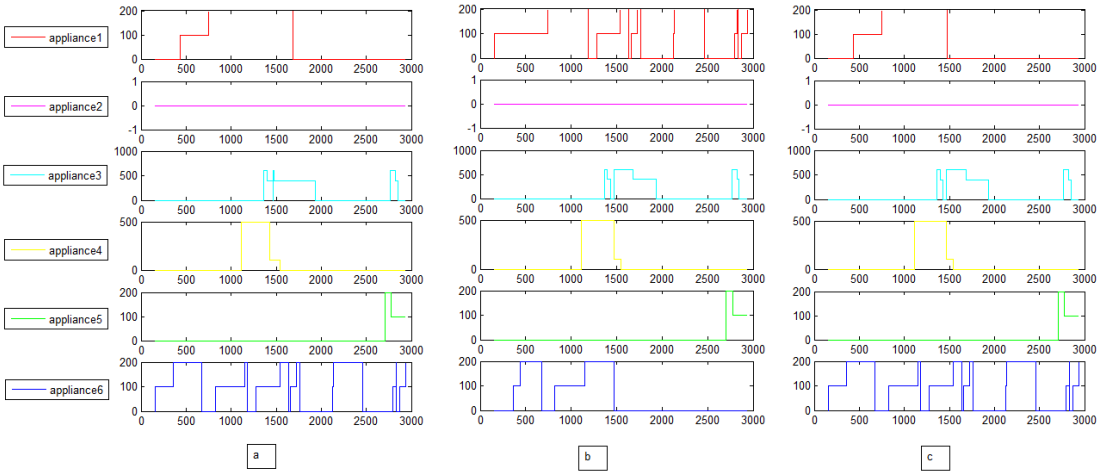

FIGURE 4.8: Appliance signatures (exponential data) a:actual signatures b:detected by power only detector c:detected by power and time detector

The second set of data which was generated based on the Gaussian time distributions are also applied to the both power only and exponential detectors, the result of this simulation shows that even for appliances which have non-Markovian behaviour, there is some improvement but it is required to investigate this algorithm for different data to see if it is reliable to apply this algorithm for non-Markovian appliances.

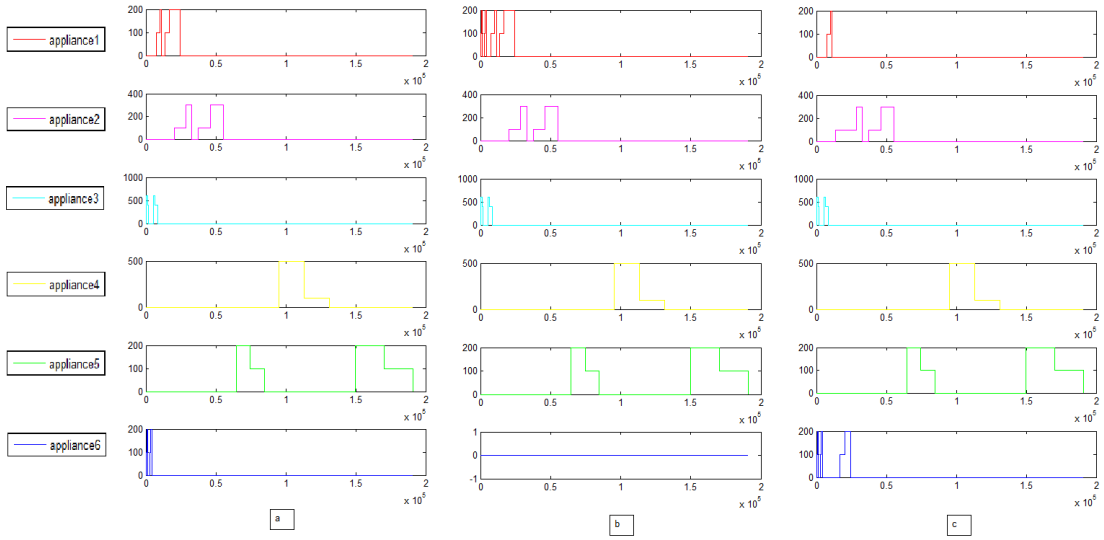

FIGURE 4.9: Appliance signatures (Gaussian data) a:actual signatures b:detected by power only detector c:detected by power and time detector

4.4.2.3 Simulation3 assumptions

To verify the accuracy of the Markovian algorithm and compare it to the power only detector, three sorts of data are generated. Three appliances with the exactly same power levels and different time distribution for each level are defined to generate these sets of data. First, it is assumed that all the appliances have the Gaussian power and exponential time distributions. Emission densities correspond to these distributions are shown in Tables 4.8 and 4.10. It is assumed that the initial state is equal to one, where all the appliances are in off level. There is no transition matrix based on state transition only, for the new detector. Transition matrix for power only detector is the same as simulation1, table 4.3. It is assumed that at each k, just one appliance state change occurs.

To generate the first set of data, at each state, based on the exponential time distribution of each appliance, which is given in 4.8, a random waiting time on the current level is generated. The minimum time has chosen and the state changes correspond to the appliance which has the minimum waiting random time, the power levels are generated based on the Gaussian distributions.

n	μ_1	μ_2	μ_3
1	1000	2000	3000
2	100	200	300
3	500	800	400

TABLE 4.8: Simulation3, time emission density of the exponential distributions, $\mathbf{\Phi}_t^{(n)}$

n	μ_1, σ_1^2	μ_2, σ_2^2	μ_3, σ_3^2
1	1000,1000	2000,2000	3000,3000
2	100,100	200,200	300,300
3	500,500	800,800	400,400

TABLE 4.9: Simulation3, time emission density of the Gaussian distributions with wide variance, $\mathbf{\Phi}_t^{(n)}$

n	μ_1, σ_1^2	μ_2, σ_2^2	μ_3, σ_3^2
1	0,10	250,10	1000,10
2	0,10	250,10	1000,10
3	0,10	250,10	1000,10

TABLE 4.10: Simulation3, power emission density, $\mathbf{\Phi}_y^{(n)}$

To generate the second set of data, the Gaussian time distributions are defined with a very wide variances ($\sigma = \mu$), where these distributions behave almost like the exponential distributions. To generate the second set of data, based on the Gaussian time and power distributions, a random life time is generated from the second level of each appliance time distribution. In the next step, the appliance with a minimum life time from the previous states is chosen, the state change is thus caused by this appliance. The power levels are generated based on the Gaussian power distributions.

The third set of data is generated based on almost deterministic signatures in time, where the Gaussian time distributions have very low variances ($\sigma = 0.01\mu$) it means dispersion is equal to 0.01. The power distribution of this data set is same as the first set. Time emission is given in Table 4.11. To generate this set of data, first, based on the Gaussian distribution in power and time, random time and power correspond to the current level for each appliance is generated, then signatures are summed in time to produce the total power consumption. The aim of applying this data is to investigate if the new detector based on the Markovian behaviour is more accurate for the appliances with the narrow Gaussian time distributions or the wide Gaussian time distributions.

n	μ_1, σ_1^2	μ_2, σ_2^2	μ_3, σ_3^2
1	1000,10	2000,20	3000,30
2	100,1	200,2	300,3
3	500,5	800,8	400,4

TABLE 4.11: Simulation3, time emission density with narrow variance, $\mathbf{\Phi}_t^{(n)}$

These sets of data are applied to the power only and Markovian detectors, the error defines and compares in next part.

4.5 Conclusion

In this chapter, two new approaches based on the VA as a solution for NIALM are proposed. The first algorithm is based on the power information and the second one is based on power and time information, which is supposed to be an improvement of the first algorithm. To see the accuracy of these algorithms, 100 simulations ($R = 100$) are executed for both algorithms. Error defines as:

$$e_k = \begin{cases} 1, & \hat{s}_k \neq s_k \geq 0 \\ 0, & otherwise, \end{cases} \tag{4.22}$$

$$Error = \sum_{k=1}^{K} \frac{e_k}{K} \tag{4.23}$$

	Gaussian distributed in time	Gaussian distributed in time	Exponential distributed in time
Dispersion	0.01	1	1
Power only detector	70%	70%	70%
Exponential detector	11%	17%	18%

TABLE 4.12: Error comparison between power only and exponential detector.

Data is generated based on Gaussian power distribution, exponential and Gaussian time distribution for dispersion equal to 0.01 and 1

As it is shown in table 4.12, the second algorithm based on the power and the time information increases the accuracy of detection when the data is generated exponentially or Gaussian in time. Considering that using more information can improve the detection, this result was predictable. Whenever the appliance signatures are generated exponentially in time which means they are memory-less, the detector which is designed based on the Markovian property could detect them reasonably. For the Gaussian generated data, which are not memory-less, this exponential detector performs almost the same as the exponentially generated data. But considering that the information from the previous state could influence the detection result for the Gaussian generated data, we need another algorithm which uses the information from the other states, thus non-Markovian based algorithm. Considering that the real data can be define more as a Gaussian distributions than an exponential distributions in time , in the next chapter a new approach will be proposed to improve this algorithm for non-Markovian appliances. It is reasonable to expect a better result from the detector which takes the information of the signatures total life time into account, and not only the time between last events.

Chapter 5

Viterbi model for non-Markovian appliances

5.1 Non-Markovian appliance

Assuming that our stochastic model is the first-order Markov model, the states at time k depend only on the previous states at time $k-1$. As was shown in the previous chapter, an appliance is Markovian if the life time distribution of different levels of that appliance are exponential. The Viterbi algorithm is optimum for these kind of appliances. But for a non-Markovian appliances which have signatures with Gaussian life time distributions for different levels, the time information from the previous states are needed. As is shown in Figure 5.1, the time that each appliance was being at a certain level before $k-1$, influences the probability of that the next event is caused by that appliance.

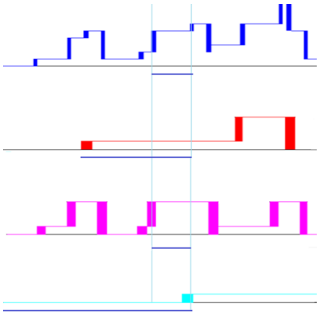

FIGURE 5.1: Life time of appliance n at time t_k, since its last state change.

In the Viterbi algorithm at each step k the metric **State**, see Chapter 3, contains the information about the winner path up to that step. In the aim of using this

information from the previous state in the Markov model, for each appliance sets one clock. Whenever the state of each appliance changes during the transition between the states the clock starts counting again from zero. This information from the clock can be shown by $\boldsymbol{\tau}_{s_k} = [\tau_{s_k}^{(1)} \dots \tau_{s_k}^{(N)}]$, k is the time step and s_k is the state, $\tau_{s_k}^{(n)}$ is the life time of the appliance n being in the state $s_k^{(n)}$ before time step k, which is shown by $\tau_k^{(n)}$ in the sake of simplicity, see Figure 5.1. The time between state $k-1$ and k is $\delta = t_k - t_{k-1}$, $\tau_{s_k}^{(n)} = \tau_{s_{k-1}}^{(n)} + \delta$.

5.1.1 Proposed non-Markovian appliance model

As was discussed before, in real world the appliances can be defined by Gaussian time distributions rather than the exponential time distributions, therefore it is desirable to modify this algorithm to the new appliance model based on the Gaussian time distributions. In this part, the Gaussian time distribution of each appliance is used to improve the previous algorithm. The emission density for non-Markovian appliances is defined by $(\mu_t)_{s_k}^{(n)}$ and $(\sigma_t)_{s_k}^{(n)}$, where $s_k^{(n)}$ is the index of the appliance's power level, see Figure 5.2.

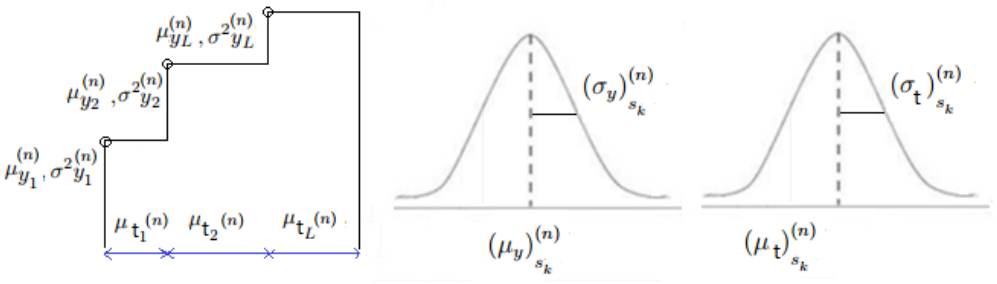

FIGURE 5.2: Non-Markovian appliance n signature

Power and time distribution of the state $s_k^{(n)}$.

5.1.2 Proposed algorithm, a Gaussian detector

As was discussed in the previous chapter, given the sequences of $\mathbf{y}, \mathbf{t}$, the sequence of $\hat{\mathbf{s}}$ could be estimated by applying equations 4.8 and 4.10. Time feature of the appliances are all independent.

The Probability Density Function (PDF) of a Gaussian distribution in time is given by:

$$g_{T_{s_k}^{(n)}}(x) = \frac{1}{(\sigma_t)_{s_k}^{(n)}\sqrt{2\pi}} \exp\left(-(x - (\mu_t)_{s_k}^{(n)})^2 / 2(\sigma_t)^{2(n)}_{s_k}\right) \tag{5.1}$$

The cumulative distribution function (CDF) of a Gaussian distribution is given by:

$$G_{T_{s_k}^{(n)}}(x) = \int_{-\infty}^{x} g_{T_{s_k}^{(n)}}(x\prime)\mathrm{d}x\prime \tag{5.2}$$

In equation 4.11, the F_n is dependent on $\boldsymbol{\tau_{k-1}}$. After taking derivative with respect to x given the PDF of a Gaussian distribution in time, and finding the contribution of the appliance $n^\star$ of the total PDF, the combined probability/PDF is:

$$Pr[s_k^*, t_k - t_{k-1}\mathrm{in}[\delta, \delta + dt)|s_{k-1}^*] = \frac{g_{n^*}(\tau_{s_{k-1}}^{(n)} + \delta)}{1 - G_{n^*}(\tau_{s_{k-1}}^{(n)} + \delta)} \prod_{n=1}^{N} \frac{(1 - G_n(\tau_{s_{k-1}}^{(n)} + \delta))}{1 - G_n(\tau_{s_{k-1}}^{(n)})} \tag{5.3}$$

5.1.2.1 Simulation4 assumptions

To verify the accuracy of the non-Markovian algorithm and compare it to the power only detector, two sets of data which are generated based on Gaussian time distribution with dispersions equal to 1 and 0.01, in simulation 3, are applied as inputs of the non-Markovian detector.

5.1.2.2 Simulation4 results

As is shown in Figures 5.4 and 5.6, for appliances with same sequence of power levels, detection result of a non-Markovian detector is much better than the power only detector, for both the narrow and the wide variances.

The non-Markovian appliances with the wide Gaussian time distribution behave more like the Markovian appliances, and the results show that the Gaussian detector is more accurate for appliances with the narrow Gaussian time distribution as was expected.

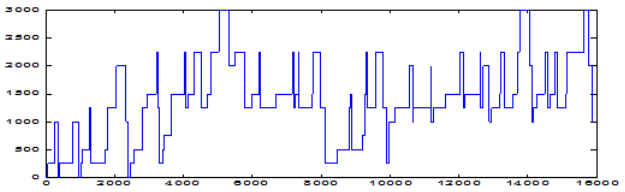

FIGURE 5.3: Total power consumption (wide Gaussian time distributions)

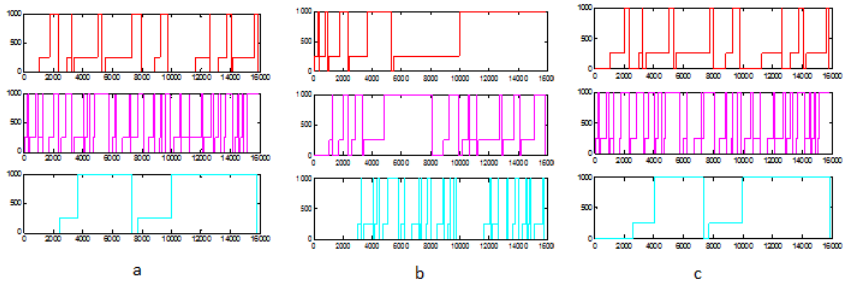

FIGURE 5.4: Appliance signatures (Gaussian data) a:actual signatures b:detected by power only detector c:detected by power and time detector

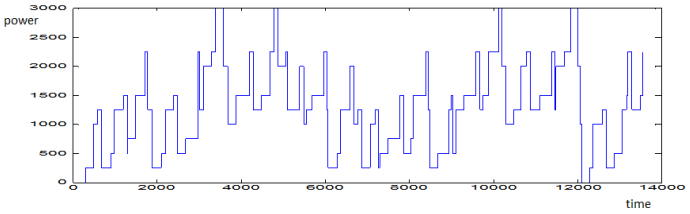

FIGURE 5.5: Total power consumption (narrow Gaussian time distributions)

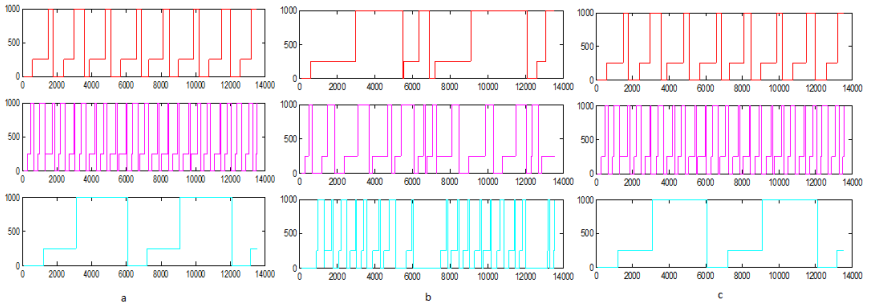

FIGURE 5.6: Appliance signatures (Gaussian data) a:actual signatures b:detected by power only detector c:detected by power and time detector

5.2 Conclusion

As is shown, proposed algorithms need different sorts of information sets: power only detector apply to Gaussian power distribution and transition matrix, the Markovian appliances detector uses the Gaussian power distributions and the exponential time distributions and the non-Markovian appliances detector needs the Gaussian power and time distributions information.

Different simulations are done to compare the accuracy of these three detectors for four groups of inputs.

Detectors are: the power only detector, the exponential detector and the Gaussian detector.

Observations are generated by: the Gaussian power distribution and the Exponential time distribution where the Gaussian detector assumes that the data generated Gaussian with dispersion= 0.01, the Gaussian power distribution and the Exponential time distribution where the Gaussian detector assumes that the data generated Gaussian with dispersion= 1, the Gaussian power distribution and the Gaussian time distribution with dispersion= 1, the Gaussian power distribution and the Gaussian time distribution with dispersion= 0.01 .
Results which is shown in Table 5.1, are based on hundred random runs for certain group of distributions, which are addressed in the description of the result figures. Three appliances with the same sequence of power level but different time features, are chosen in all these simulations.
As is shown in Figure 5.7, the power only detector always chooses the first appliance, in the case that data is generated based on the Gaussian power distributions and the exponential time distributions. It means that it can not distinguish the appliances with the same power level as was shown before. The Markovian appliances detector can find the optimum path, error is reduced to 18%, see Table 5.1. The non-Markovian detector with the wide variance assumption, equal to the mean have a good result close to the Markovian appliance detector result. It is because of the similar behaviour of the exponential distributions and the wide Gaussian distributions.
As is shown in Figure 5.8, the power only detector can not distinguish the appliances, for generated data based on the Gaussian power and time distribution. The Markovian appliances detector performs better than the non-Markovian appliances detector, because data is generated based on the wide Gaussian time distribution, thus detector has the same assumption as the generated data. The Markovian detector can also detect this data with a good accuracy but still the result is not as good as the non-Markovian appliances detector.
As is shown in Figure 5.10, the Gaussian detector with narrow variance, can not detect anything when the data is generated exponentially. The detector assumes that data is generated by the narrow Gaussian distribution which is very different from reality. The exponential detector has a good performance as was expected.

	Gaussian distributed	Gaussian distributed	Exponentially distributed	Exponentially distributed
Dispersion	0.01	1	0.01	1
Power only detector	70%	70%	70%	70%
Exponential detector	11%	17%	18%	18%
Gaussian detector	1%	15%	99%	24%

TABLE 5.1: Error comparison between different detectors

Data is generated based on the Gaussian power distributions, the exponential and the Gaussian time distributions for dispersion of 0.01 and 1

The non-Markovian detector optimally and the Markovian detector reasonably perform, where data is generated based on the narrow Gaussian distribution. For exponential detector also the error decreased because the random variables are chosen close to the mean of the exponential distributions. The summery of these simulations is shown by Figure 5.11, it shows the performance of the Markovian and non-Markovian appliances detectors for input data generated by different distributions. The Markovian appliances detector has the best performance whenever the data is generated by narrow Gaussian distribution which is more close to the reality.

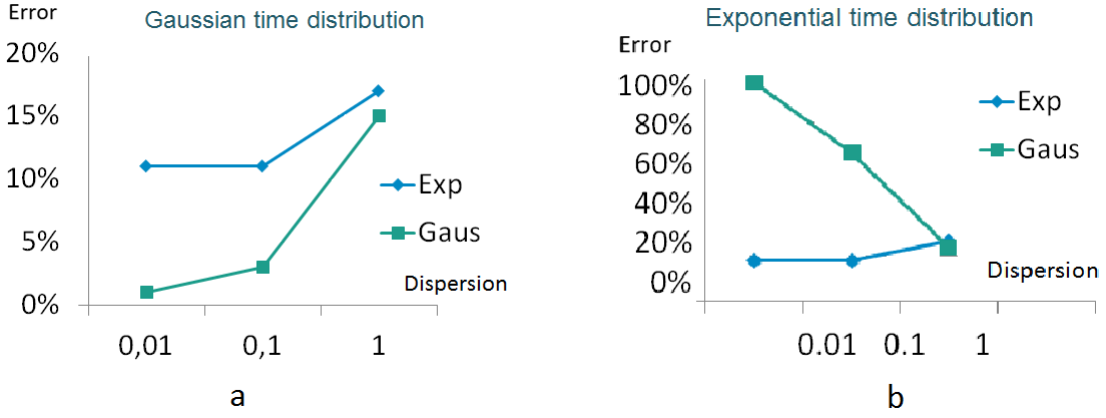

FIGURE 5.11: Performance diagram of the exponential detector and the Gaussian detector.

a.data generated based on the Gaussian time distributions, b.data generated based on exponential time distributions

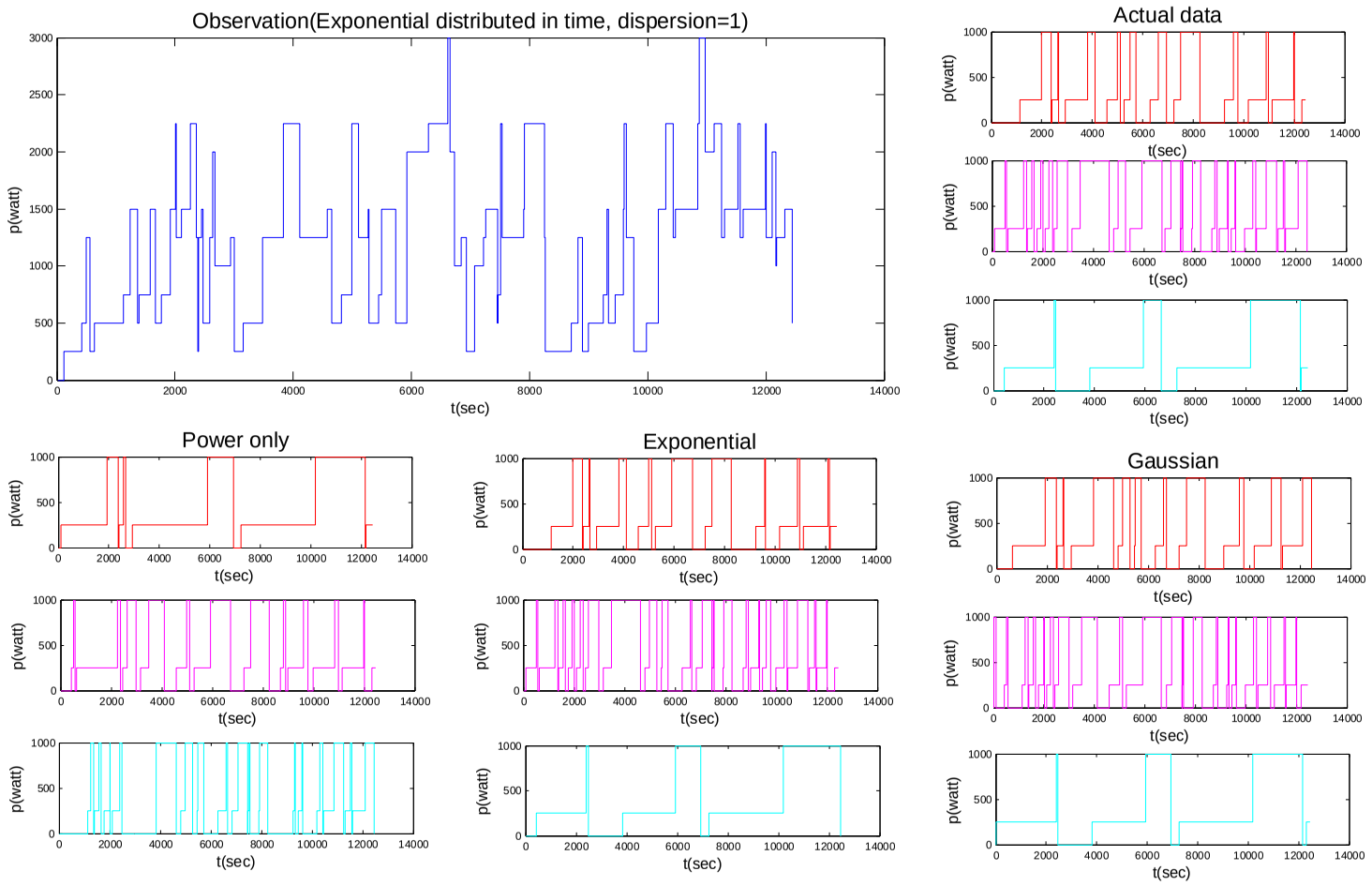

FIGURE 5.7: Simulation result: data is generated based on the Gaussian power distributions and the exponential time distributions, dispersion= 1.

Inputs: tables 4.2, 4.10 and 4.8.
Red: appliance1, magenta:appliance2, cyan:appliance3. Three detectors results are shown (Gaussian detector assumes that data is generated based on the Gaussian time distributions with dispersion= 1)

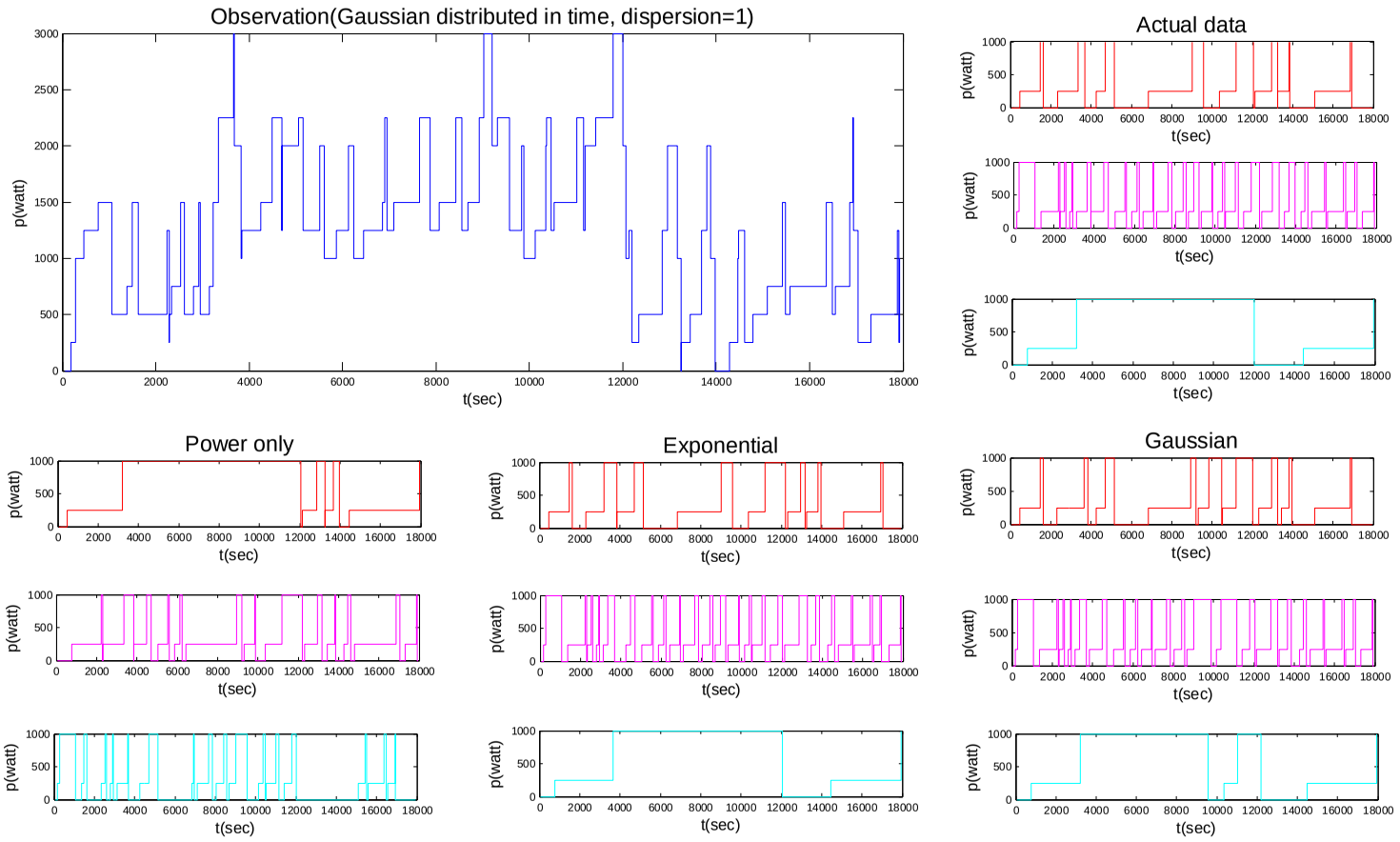

FIGURE 5.8: Simulation result: data is generated based on the Gaussian power distributions and the exponential time distributions, dispersion= 1.

Inputs: tables 4.2, 4.10 and 4.9.
Red: appliance1, magenta:appliance2, cyan:appliance3. Three detectors results are shown (Gaussian detector assumes that data is generated based on Gaussian time distribution with dispersion= 1)

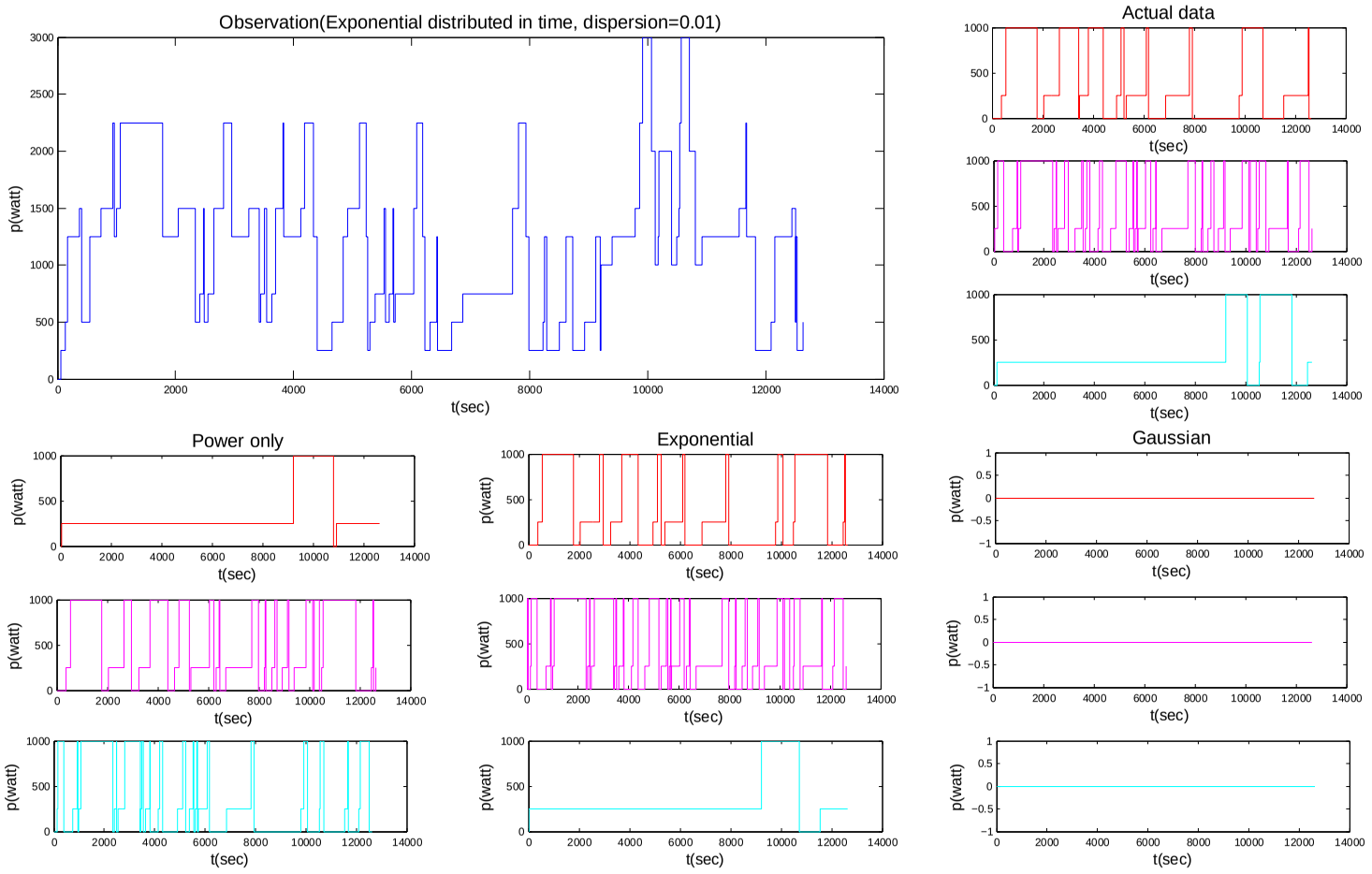

FIGURE 5.9: Simulation result: data is generated based on the Gaussian power distributions and the exponential time distributions, dispersion= 0.01.

Inputs: tables 4.2, 4.10 and 4.8.
Red: appliance1, magenta:appliance2, cyan:appliance3. Three detectors results are shown (Gaussian detector assumes that data is generated based on the Gaussian time distributions with dispersion= 0.01)

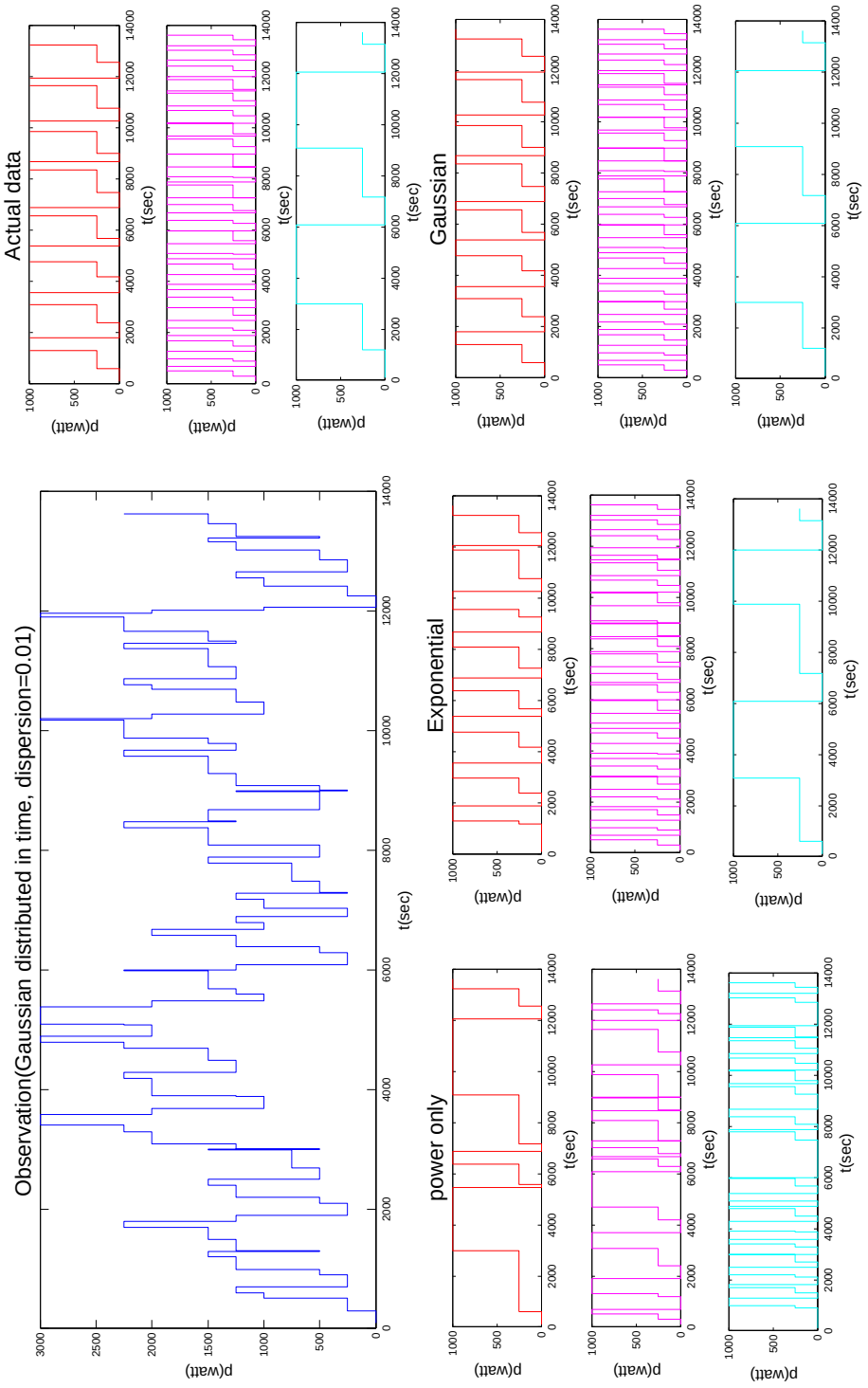

FIGURE 5.10: Simulation result: data is generated based on the Gaussian power distributions and the Gaussian time distributions, dispersion= 0.01.
Inputs: tables 4.2, 4.10 and 4.11.
Red: appliance1, magenta:appliance2, cyan:appliance3. Three detectors results are shown (Gaussian detector assumes that data is generated based on the Gaussian time distributions with dispersion= 0.01)

Chapter 6

Conclusion and future works

6.1 Conclusion

The goal of this thesis was to find a new approach to NIALM, in the aim of efficient usage of available information. The Viterbi as an algorithm which provides a strong correcting capability, relatively small memory usage and flexible structure to bring different information into account, was looked very promising to find the optimum solution using information as efficient as possible.

Three proposed algorithms are: power only detector , the Markovian appliances detector and the non-Markovian appliances detector. These algorithms, based on the available appliance signature information, are applied for different levels of information availability.

The power only detector finds an optimum path by applying the Gaussian power distribution and transition matrix. Most of the available algorithms use only Gaussian power distribution. The combination of these distribution information, the transition matrix information and the Viterbi algorithm gives the opportunity of finding the optimum path, hence, a very better result compare to the available algorithm. The Markovian appliances detector uses the Gaussian power distribution and the exponential time distribution. The extra time information plus the previously mentioned information leads to an even better result but not the best yet. The non-Markovian appliances detector needs the Gaussian power and time distributions information. This detector uses the appliance model which is very realistic and provides as much information as possible, therefore the result is a lot better due to the efficient usage of information and finding the optimum path.

This means that this approach is applicable for different databases and uses the given information as efficient as possible to find the optimum path.

6.2 Future works

This thesis was focused on the information given by the event detector, which are the power and time, assuming that the appliances database provides the signatures features only. But it is also possible to gather some information about users behaviour like usage time of the appliances, based on seasons, week or weekend, day or night and some other information like the probability of that the certain group of appliances could or could not work simultaneously, set a threshold on power usage and any other information which can affect the transition probability between the states in a purpose of accuracy.

The other important improvement possibility to this algorithm, which is mentioned in the introduction, is related to the training part. It is possible to proposed an algorithm to provide an automatic setup (AS-NIALM). This helps the algorithm to be independent of the manual setup and also provides the opportunity to self update and make the appliance distribution more accurate over the time. The other advantage is the possibility of recognizing the new device and defining the proper distributions to its power sequence and corresponding time.

Bibliography

[1] G.Hart. "nonintrusive appliance load monitoring". *Proceedings of the IEEE*, 80:1870–1891, 1992.

[2] J. Z. Kolter, Siddharth Batra, and Andrew Y. Ng. Energy disaggregation via discriminative sparse coding. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1153–1161. Curran Associates, Inc., 2010. URL http://papers.nips.cc/paper/4054-energy-disaggregation-via-discriminative-sparse-coding.pdf.

[3] H.Pihala. "non-intrusive appliance load monitoring system based on a modern kwh-meter". Technical Research Centre of Finland, 1998. ISBN 9513852482, 9789513852481.

[4] R.L.Tweedie S.P.Meyn. *"Markov Chains and Stochastic Stability"*. 1993.

[5] T.Olofsson M.Johansson. "bayesian model selection for markov, hidden markov, and multinomial models". *Proceedings of the IEEE*, 14:129–132, 2007.

[6] G.R.Nudd M.S.Ryan. "the viterbi algorithm". 1993.

[7] M.Weal A.Rogers O.Parson, S.Ghosh. " non-intrusive load monitoring using prior models of general appliance types". *Proceedings of the Twenty-sixed conference on Artifical intelligence*, pages 356–362, 2012.

[8] K. Roth M. Zeifman. "viterbi algorithm with sparse transitions (vast) for nonintrusive load monitoring". 2011.