A Smart Home – Analysis and Scheduling

Camiel van Altenborg (4091515)
Jente Zandstra (4084888)

June 22, 2015
Abstract

In order to cope with various supply sources and loads in an autonomous micro-grid, a system of sensors and control units was developed. The goals were to cope efficiently in the event of an energy supply shortage, and to prevent such events as much as possible. To achieve these goals the aim was to manage energy in the system as efficiently as possible. The system consists of a collection of sensor nodes, a control unit and an analysis unit. The sensor nodes monitor power flow and collect data from the outlets. The main control unit manages the power flow and prevents shortages, and relays information between the nodes and the data analysis unit. The data analysis unit uses historical data collected from the nodes to make predictions on supply and demand and to make scheduling decisions.

This thesis is concerned with the data analysis unit. Different load signatures and usage patterns are discussed. Subsequently a theory is proposed to extract the desired patterns. Successively, an implementation was made using Python. Afterwards, the theoretical methodology and the implementation schemes are evaluated, followed by recommendations for future work.
Contents

1 Introduction 5
  1.1 Current domestic power grid 5
  1.2 Other work 6
  1.3 Proposed microgrid 6
  1.4 Outline of this thesis 7

2 Problem definition (overall) 8
  2.1 Overview 8
    2.1.1 Reducing total cost of energy 8
    2.1.2 Reducing dependency on mains power 8
    2.1.3 User interaction 9
  2.2 Sub-problems 10

3 Top-level design 12
  3.1 MCU 12
  3.2 Nodes 13
  3.3 Data analysis 14
Preface

This thesis has been written with respect to the Bachelor Graduation Project of the Electrical Engineering program. It is written for the members of the jury of the thesis defense, and for anyone who is interested to read about microgrids.

Two theses are written for this project – one thesis focuses more on the hardware, and describes the decision-making in the microgrid, and the other thesis focuses more on the software and mathematical algorithms. This thesis is about the analysis and scheduling software.

We would like to acknowledge our supervisors Dr. Mathijs de Weerdt, Dr. Matthijs Spaan and Dr. Germán Morales-España for their help with writing our thesis. Without them, our thesis would look less good.
Chapter 1

Introduction

1.1 Current domestic power grid

The current domestic power grid has been around for a while. It serves its purpose – providing power to the sockets in the whole house – well. You can plug in an appliance, like a TV, a refrigerator or a washing machine, in an available socket, which is powered by the grid, and the appliance works.

The current grid, however, has pretty basic functionality – providing power from the main grid to the sockets – and therefore has many areas to improve in. One of these areas is the use of distributed energy resources (DERs), e.g. solar panels. The fossil fuels are being depleted, so other energy sources are needed. Many believe that the DERs are the future – sustainable energy from the sun, the wind and moving water are inexhaustible, and can potentially replace the fossil fuels. Before this can happen, changes need to be made to the current power grid. Solar panels can already be connected to the current grid, but there is room for improvement.

Another area of improvement is the ‘brain’ of the grid – the current grid just provides power to the sockets, even to the sockets who aren’t in use at the moment. A lot can be gained if the grid only powers the sockets currently used.
1.2 Other work

There are several companies who are trying to improve the current grid as well, or try to reduce the energy cost of the users.

Nest has created the Nest Learning Thermostat, which, as the name suggests, is a smart thermostat. Just by using it, it learns the user's schedule and preferences. Moreover, Nest has some other functionality, e.g., sensing if someone is home. Nest claims to cut 20% of the user's heating and cooling bill. Nest and the proposed microgrid have a few things in common: both are self-learning and both reduce the energy and the energy costs.

Toon has created a smart thermostat too, which shows the energy usage and the costs of this usage. Just like Nest’s thermostat, it is self-learning, and reduces the energy costs. Toon, however, has two extra, interesting features. It can control some sockets, plugging it on or off, just like the proposed microgrid. Furthermore, it works with the user’s solar panels, which is a main function of the proposed microgrid.

Smappee has created a device, which is more than a smart thermostat. It shows the total energy usage and costs, but also the energy usage and costs per device. Besides monitoring the usage of a device, it can also switch off devices, both automatically and according to the user. Moreover, it is compatible with solar panels.

1.3 Proposed microgrid

This thesis proposes a smart microgrid, which can replace the current domestic grid. It has all the functionality of the current grid, and more. DERs can be connected to it, and it decreases the dependency on the main grid – the proposed microgrid can operate without being connected to the main grid. It also reduces the energy cost through scheduling, and it only powers the sockets currently in use.

Our vision is that the proposed microgrid replaces the current domestic grid – it can be installed in residential buildings, e.g., houses and farms. Besides the consumer market, it can also be sold to do-it-yourself stores, e.g., Gamma.
1.4 Outline of this thesis

The thesis starts with a thorough definition of the problem in Chapter 2, in which the problem is divided in sub-problems, some of which are discussed in this thesis. Thereafter, our top-level design is discussed in Chapter 3, in which the microgrid is divided into different units. Then, in Chapter 4, some ethical considerations are discussed, followed by Chapter 5, in which the sub-problems of Chapter 3 specific to this thesis are discussed in more detail. In Chapter 6, theoretical background, needed for the data analysis, is given. The implementation of the system is described in Chapter 7, and the evaluation of its effectiveness in Chapter 8. Chapter 9 provides a more theoretical discussion, and finally Chapter 10 gives recommendations for future work.
Chapter 2

Problem definition (overall)

The purpose of the system is twofold: to reduce the total cost of energy for the user, and to reduce the user’s dependency on the electrical mains. The following section will provide an overview of the means to achieve these goals, followed by a discussion of the system’s interaction with the user. With this overview in mind, we can then identify sub-problems: a concrete list of engineering challenges to be solved.

2.1 Overview

2.1.1 Reducing total cost of energy

There are two principal means of reducing energy costs: reducing total energy consumption, or moving consumption to hours with cheaper rates. Our system will focus on the latter, by scheduling loads such as washing machines or dishwashers to run when rates are low.

2.1.2 Reducing dependency on mains power

Reducing dependency on the mains means facilitating independent (islanded) operation of the microgrid. Concretely, the system should ensure balance of power in the microgrid at all times, preferably without connecting to the mains. The means to achieve this can be divided into static and dynamic functions.
Statically, maintaining balance of power simply means monitoring power consumption and production, and taking action when there is a discrepancy. This action could be to disconnect loads (in the event of a shortage) or sources (in the event of a surplus) from the microgrid. However, this is a highly undesirable outcome and should only be used as a last resort. Load shedding means forcing shut-down of certain appliances, which is obviously inconvenient to the user. Source shedding is wasteful, especially in case of sustainable sources; these are simply not always available, so their potential should be fully used when they are.

A much more attractive static measure, then, is to simply connect the microgrid to the mains. These can then be used as a power source (providing power in the event of a shortage) or power sink (absorbing power in the event of a surplus). In both cases, the mains are assumed to have an unlimited capacity, at least when compared to the small grids our system is designed for.

Clearly, although static measures to maintain balance of power are unquestionably important, they alone cannot reduce the microgrid's dependency on the mains – at least not without introducing considerable inconvenience to the user. Therefore, dynamic measures should be taken, in order to make it less likely that static control actions will be necessary in the first place.

Here, too, the system will make use of scheduling. By monitoring production and consumption in the microgrid over a longer period of time, predictions can be made about the production and consumption in the near future. These can then be used to make scheduling decisions (for example, whether to switch on a washing machine now, or in two hours’ time). By synchronising peaks in consumption with peaks in production, and making effective use of buffers to prepare for upcoming variations, this will make it less likely that discrepancies between consumption and production will arise.

2.1.3 User interaction

One of our principles in designing the system is that manual configuration by the user should be possible but not necessary. The system should be able to run autonomously without any input from the user, but the user should still be able to manually steer its operation – through a mobile app, for example, or a desktop program – so that it best serves his or her needs. More concretely, the user might specify that a buffer of a given capacity is plugged into a certain socket, or that the appliance plugged into another socket should be switched on at some point in the next 24 hours (while leaving it to the system to figure out exactly when). Forcing islanded mode (so that the microgrid will never be connected to the mains, even if the only alternative is load or source shedding) is another example of an option available to the user.
Ideally, the user should be able to give commands to the system from anywhere. When it comes to the other direction – output of information to the user – privacy is an important consideration, given the amount of data our system will collect about the user’s living patterns.

2.2 Sub-problems

Given the overview above, we can identify several engineering challenges that need to be solved in order for the system to work as described.

1. *Frequency control:* Maintaining a stable frequency in an AC microgrid in islanded mode.
2. *Mains connection control:* Reconnecting the microgrid to the mains after islanded operation, without phase discrepancies or voltage surges.
3. *Sensing:* Sensing the power consumption of a load (or power production of a source) in an AC grid, without affecting the load’s behaviour.
4. *Switching:* Disconnecting sockets from the grid, or reconnecting them to it, when the system decides this is necessary.
5. *Analysis and prediction:* Detecting meaningful patterns in historical data, and estimating future behaviour based on those patterns.
6. *Scheduling:* Deciding which dynamic control actions are necessary, based on the forecast from point 5.
7. *Static decision-making:* Deciding to apply static and dynamic control actions, prioritising one over the other if necessary.
8. *User input:* Receiving user input and processing it.
9. *Communication:* Enabling the different parts of the system – sensors, switches, controller and analysis module (points 3, 4, 7 and 5, respectively) to relay information to each other.

Because the available time and resources for this project are limited, we have to choose which of these problems we are going to solve; it is not feasible to solve them all. We have decided to focus on those solutions that can be relatively easily implemented – and demonstrated to work – with a minimum of hardware. With that criterion in mind, the choices we have made are as follows:

- In this thesis ("Analysis and Scheduling"), we discuss the solutions to points 5 and 6.
• In the "MCU and Nodes" thesis, we discuss the solutions to points 3, 4, 7 and 9.

• The solutions to points 1, 2 and 8 are left for future work. However, points 1 and 8 will be (briefly) touched upon in the "MCU and Nodes" thesis.
Chapter 3

Top-level design

In this chapter, the general design of our system will be described. Note that this is meant as an outline of the final system, that is, the system that would be build given enough time and resources – and not the system that will actually be build within the project’s timespan.

This final system will consist of three principal components: a main control unit (MCU), a data analysis unit (DAU), and node hardware. Each of these components will be discussed in a subsection.

3.1 MCU

The MCU is the controller of the system. It is responsible for the static functionality as described in section 2.1.2. The MCU is fully static, in other words, memoryless: its decisions are based entirely on the current values of the input signals it receives, not on their history.

It sends switching commands to the nodes, instructing them to open or close their relays. These commands are the outcome of a decision-making process, for which it receives input from two sources: the nodes themselves, and the data analysis unit.

From each node, the MCU receives information on the status of that node: the measured power flow into or out of the socket, the readouts of any other sensors connected to the node, and possible user-specified parameters for the node.
From the data analysis unit, the MCU receives basic commands. These will instruct it to open or close specific nodes’ relays.

However, the MCU can also operate without input from data analysis, relying only on input from the nodes. This is necessary for two reasons. Firstly, the system will start without any data to analyse, and it will take some time to gather enough data for meaningful pattern detection; secondly, the system’s basic operation should not depend on the data analysis, which is relatively complicated and error-prone compared to the MCU.

Without input from data analysis, all the MCU will do is maintain balance of power by the static methods mentioned in section 2.1.2. Even when the data analysis unit does send commands, the MCU may decide to ignore these if they conflict with balance-of-power considerations.

### 3.2 Nodes

At each socket connected to the microgrid, there is a small set of hardware, referred to as a node. Each node contains a microcontroller, with at least two components connected to it: an electricity meter and a relay. Additional sensors may also be connected.

The electricity meter measures how much power is flowing into or out of the node’s socket. The relay can disconnect the socket from the microgrid; the microcontroller will open or close this relay when commanded to by the MCU.

Other sensors that may be connected include thermometers, light sensors, and motion detectors. These can provide very useful information for analysis – for example, a gradually decreasing light level might mean that the production of solar cells will soon drop to zero, and motion detectors might be used to sense whether or not anyone is home.

It is important to note that all nodes are, in principle, interchangeable; a load, a source or a buffer may be plugged into any given node’s socket. The exception is the mains connection node, which requires specialised switching equipment to prevent harmful transients when connecting to or disconnecting from the mains.
3.3 Data analysis

The data analysis unit is the most sophisticated part of our system. It is responsible for the dynamic functionality described in section 2.1.2, as well as all the functionality outlined in section 2.1.1.

The data analysis unit receives data from all nodes in the system. These data are logged and analysed in two principal ways. Firstly, the data analysis unit detects patterns over time in individual signals; in this way, it might teach itself the day-night cycle and other temporal patterns. Secondly, it detects correlations between different signals; for example, a positive readout from a motion detector might strongly correlate with a sharp rise in power consumption a few minutes later.

Based on this combination of time patterns and cross-correlation between signals, the data analysis unit makes predictions about the power production and consumption in the coming hours. These predictions, in turn, serve as input for a decision-making process: the unit might decide that now is a good time to start charging up a buffer, or to connect a load that has been set to switch on within a given time period (as mentioned in section 2.1.1). Concretely, this translates to instructions to the MCU to open or close specific nodes’ relays. In this way, a feedback loop of information is established (nodes \(\rightarrow\) data analysis \(\rightarrow\) MCU \(\rightarrow\) nodes).
Chapter 4

Ethical considerations

The primary societal effect our system is likely to cause, is an increase in the use of distributed energy resources (DERs), such as solar panels, small wind turbines, and small (bio)gas generators – as well as an increase in local energy storage. This will contribute to the sustainability transition, and empower consumers to provide their own electrical energy, instead of being completely dependent on the main grid. The social values the system will serve are thus sustainability and self-reliance.

It is mostly consumers and DER producers who will benefit from the effects of our system. On the other hand, the interests of energy companies and transmission network operators (TSOs) might be hurt: energy companies will be able to sell less energy, and TSOs will end up with a grid that is harder to control because of the greater amount of local generation. The interests of energy companies are not a very large factor in our ethical considerations, as the conservation and sustainable generation of energy is far more important than the profit of any company anyway. On the other hand, the interests of TSOs are somewhat more of a concern. However, since the transition to a more DER-heavy grid is more or less inevitable if the sustainability transition is to succeed – introduction of our system will merely speed up the process, as it were – this is not a big ethical objection for us.

Although we will do whatever possible, to adhere to the highest standards of sustainable production, it is inevitable that production of our system will cause some damage to the environment. However, this can be justified through utilitarian ethics, as we expect that the aforementioned sustainability gains, resulting from the use of our system, will outweigh the damage done by its production.

Naturally, we will adhere rigorously to electrical safety standards in the design,
production, and delivery of our system. After all, according to Kant’s deontological ethics, we have the moral duty to make our system as safe as we would want the system designed by others in our own homes to be.

The system will be produced in industrialised countries that are relatively far along in the sustainability transition, such as Germany or Denmark. With the ethics of responsibility in mind, we make a conscious decision to avoid the low wages, corruption, and questionable labour practices in "cheap" production countries such as China – without hiding behind arguments along the lines of 'if we don’t do it, others will'. This will increase production costs, of course, but since our target market are small businesses and relatively affluent consumers, we expect to be able to afford these extra costs. Besides, with the considerable recent interest in "local production" – especially among people who are also interested in sustainability – this might even be an attractive sales argument!

The values of sustainability and self-reliance, mentioned above, will also be reflected in our corporate policy. "Sustainability" will be translated into mechanisms to take the environmental effects of important corporate decisions into account, and "self-reliance" into a hands-off management style that leaves a considerable amount of freedom and responsibility for individual employees.

To increase cohesion and solidarity within the company, we will pay the same salaries to employees with equal or similar responsibilities; we will not hand out performance bonuses. This lack of financial incentive might decrease productivity slightly, but we find it more important to avoid an atmosphere of competition among colleagues.

With regards to hiring policy, it is important that we should hire people for their talents and expertise, without regard for race or gender. This is mostly based on virtue ethics: it is a virtue to give everyone a fair chance. To ensure that this is carried out in practice, our annual reports will include an "equal opportunity section" which comments on the actual race and gender statistics of the new employees taken on the past year, as well as long-term patterns in these figures.
Chapter 5

Problem definition (analysis and scheduling)

Scheduling of devices and allocation of resources to storage units can be implemented using historical data. If specific devices having smart home interface possibilities are present, scheduling can be implemented in their operation. This will facilitate balancing supply and demand, as well as increasing efficiency in the operation of certain devices. This is possible for instance when a thermostat is controlled using information about the behaviour of the user.

Detection of loads can be achieved by power measurement, detection of events, and clustering and matching these events, and equipment identification. Correlations of consumption and peak load with time on a monthly scale can, if extended over periods of similar weather, reveal trends, up or down, that indicate in a rough way the impact of or need for conservation activity or load control [1]. The combination of a high spatial and temporal resolution opens possibilities for accurate specification of load and usage characteristics.

Load signatures [2] and appliance usage patterns will contain detectable patterns. Recognising and anticipating these patterns will allow significant improvement of the efficiency in the use and allocation of power [1]. Identifying load signatures can yield estimates for the flow of energy through the system. This would result in the ability to detect meaningful patterns in historical data, and estimating future behaviour, as proposed in 2.2. In combination with other sensor data usage patterns can be analysed, and used to improve predictive value. Loads connected at nodes will behave according to a usage pattern, either
- constantly on
- on/off states
- single cycle
- multiple cycles
- multi-state

Figure 5.1: Examples of load signature patterns. From top to bottom: constantly on, on/off cycle, single cycle, multiple cycles, and a multi-state type cycle.

Examples of these different patterns are shown in figure 5.1. When a load confirms to a constantly on usage pattern, there is only one state corresponding to the load. Hence there is no way to control the load to increase efficiency without disabling it. Contrastingly, energy requirements will be constant over time, reducing the complexity for anticipating the power signature for this type of load.

Loads that have on/off signatures have a constant impedance, but may have a time variant usage pattern. Here a further distinction can be made according to whether the duration for the on state is time is irregular or constant. A lamp connected to a timer might always be on for a specific duration.

A reading lamp, by contrast, might be on for irregular periods of time. These loads are a special case of single cycle loads. These loads are function-specific appliances that run through a predefined cycle each use. The load signature might have multiple states during the cycle, although the pattern is recurrent every time the cycle is initiated. An example for such an application might be
a dish washer, or clothes dryer. This class in turn is part of a more general type of cycling loads: those with multiple cycle options.

These appliances perform a function with corresponding invariant load signatures. However, such loads may have multiple cycle options, each with individual signatures. A common example might be a washing machine. Combinations of types can also be imagined, such as a microwave with a constant load signature, but configurable output power for different modes of operation. Lastly, continuously adjustable loads can be imagined, as for instance dimmers or power tools or other adjustable equipment. Ultimately, all load signatures can be described as multi-state.

Insofar as the definition of a state is allowed to extend to infinitesimal state durations, a multi-state load is analogous to the criterion of continuity of a function [3]. Analogously, to the extent that state durations for on-off state appliances can be considered as a discrete set it could be argued that this load is in fact of multi-state type.

Furthermore, patterns could be extracted from use time data, and correlated sensor information. Many appliances in both industrial and domestic situations are either activated at or around specific times, or active during a specific segment of a day. Even though accuracy and precision may vary, the information can be used to improve scheduling even if consistency of activation times and durations is moderate. It must be noted that when considering such usage frequencies skewing may occur due to for instance seasonal modulations in these patterns. Compensating for such drift patterns must be considered, as well as the possible consequences for the necessary and sufficient conditions for application of analysis methods. Finally, some guage must be provided on the computational requirements to estimate the range of analyses possible for certain constraints on the update interval and data set size.

Thus, given the availability of a large record of history of environmental data the goal is to extract relevant handles, from which to construct predictions of future behaviour and facilitate scheduling of applications connected to the system. To do so the data must be sanitized, appropriate algorithms must be applied to extract relevant handles, and these handles must be converted into an appropriate handling scheme. Sanitizing the data means that influences of errors and noise are minimized. Secondly, finding appropriate algorithms implies that relevant patterns can be extracted. Finally, translating handles into an appropriate handling scheme requires that the results of analysis are translated into processible prediction and scheduling decisions.
Chapter 6

Theoretical foundations

Sampled measurements will be gathered from variables relevant for estimation of supply and demand. The resulting data set is a bounded and discrete approximation of the surroundings, plus noise introduced by the measurement. From these sets load signatures are extracted by correlating sensor values. Furthermore, the collection of sets of sensor values can be correlated for extraction of patterns of use. The collection of sets, hereafter referred to as the data set (sets and variables thereof will be referred to as data) therefore is itself a discrete set of states, or state space. Assuming the existence of load signatures and patterns of use is reasonable, albeit under certain restrictions.

6.1 Pattern recognition

The data set is a state space whose states are determined by:

- the nodes \( \mathcal{R} \)
- the sensor values \( J_k \) at each node \( k \in \mathcal{R} \)
- the range of the sensor values \( V \)
- set of transitions

In order for patterns to exist and be detectable some restrictions must be imposed on the state space. First and foremost, the possible states may not depend
on the number of state transitions. This restriction \[3\] implies that the probability that the next state is some state \(j\) (i.e., \(P(X_1 = j)\)), is the sum of the initial probabilities for all states and the probabilities of transitioning from these states to state \(j\) (\(P(X_1 = j) = \sum_{i \in S} P(X_0 = i, X_1 = j)\)). Formally, it ensures that there exist a probability triple \((\Omega, \mathcal{F}, P)\) and random variables \(X_1, X_2, \cdots\) defined on \((\Omega, \mathcal{F}, P)\) such that for every sequence of states \(i_0, \cdots, i_n\):

\[
P(X_0 = i_0, X_1 = i_1, \cdots, X_n = i_n) = v_i p_{i_0 i_1} \cdots p_{i_{n-1} i_n} \tag{6.1}
\]

For \(i_0, \cdots, i_n\) states, \(v_i\) the probability for \(i\) being the initial state, and \(p_{ij}\) the probability of a state transition from \(i\) to \(j\). This criterion is the Markov criterion. This criterion may be relaxed to depend on \(N\) preceding states, forming an \(N^{th}\) order Markov chain. The duration of a state may vary according to the function of the connected appliance. For instance when a reading lamp is connected, the duration of the \(on\) state may vary significantly. To account for these effects two options are available. If the order of the Markov chain is such that the duration of a constant level is accounted for, the duration will be anticipated to some degree. This is achieved by consideration of sufficient preceding states. Otherwise, the temporal aspect can be omitted from the transition matrix.

Moreover, patterns must be recurrent if predictive value is to be obtained. This requires that the probability of reaching state \(i\) after some number \(\tau \in \mathbb{N}\) of state changes from state \(i\) is 1 for some \(\tau\). Intuitively; recurrence means that a state is certain to be returned to within some appropriate interval (to make this explanation properly recurrent; it means all states must be recurring).

These restrictions suggest that the state space for all nodes is static, at least for the order of the Markov chain considered. That is, the propagation through the state space is governed by an invariant probability distribution. Rather, that means that the probability distribution for state transitions is equal to the probability distribution relative to earlier initial states. In more formal terms this means there is a stationary distribution of possible state transitions \(\{\pi_i\}\) for which the transition matrix \([p]\) is an identity operator: \([\pi][p]^n = [\pi]\) for any \(n \in \mathbb{N}\). As suggested by the data set characteristics however, time-information is available as a result from the sampling of sensor values at regular intervals. Although state transitions should therefore be interpreted independent of the time between transitions, there is information about the duration of specific states.

The duration of a state may vary according to the function of the connected appliance. For instance when a reading lamp is connected, the duration of the \(on\) state may vary significantly. To account for these effects two options are available. Firstly, if the order of the Markov chain is such that the duration of a constant level is accounted for, the duration will be anticipated to some degree.
This is achieved by consideration of sufficient preceding states. This solution preserves the irreducibility of the Markov chain. Otherwise, the temporal aspect can be omitted from the transition matrix. This way the Markov chain preserves the property of being stationary. An additional advantage is that the order of the Markov chain can be restricted. This solution implies that only state transitions to non-identical state transitions are considered in the transition matrix. This way state transitions are deemed invariant with respect to time decimation.

If states are assigned an expected duration, an expected value and a variance can be determined for the non-identical state transition. The state transition probabilities are then stationary. This can explained by interpreting the transition sequence as a random walk between states. If some pattern is known, the assumption is that a state change (from one state to another) will occur following the stationary probability distribution of conditional state transitions given an initial state $v_i$.

An expected value and a variance can be obtained for the duration of a state. This is analogous to incremental changes in transition probabilities in a Markov chain of sufficient order. Thus, allowing a certain measure of variance on the estimate of A sequence $\{p_{ij}(\tau)\}_{n<\mathbf{N}}$ of $\mathbf{N}$ transition probabilities will converge to the stationary probability $[\pi]^{-1}$ as $\tau \rightarrow T$ in a non-linear fashion, as $\lim_{n \rightarrow \infty} P^n_{ii} = 0, P^1_{ii} \approx 1$. That is to say that the probability that a state change occurs diminishes as $\tau > \mu(v^{(i)}_t)$ as the likelihood for a state change increases when the duration for a state exceeds the expected duration.

the time a non-identical state change will occur given some initial sequence (corresponding to the order) of states, the variance is relegated to an uncertainty in the time domain. Resultingly, a stationary distribution is used, at the expense of not having irreducible Markov chains.

These efforts yield a stationary transition distribution of the probabilities for transitioning from any initial state to any $\mathbf{N}$ other, where $\mathbf{N}$ is the order of the Markov chain. Separately, expected wait times are obtained corresponding to when the state change is expected to occur. When combined these describe the expected pattern for a load. Combining the transition probabilities and duration information for every node results in a specification of the expected power demand for $\mathbf{N}$ states at every node, and the expected durations between traversions.

\footnote{It should be noted that having an expected value and a variance implies that the expected value does not actually coincide with the convergence to the stationary distribution $[\pi]$. This is because the normal distribution's expected value lies at the 50\% probability point and therefore does not exclude the identity state change at that point. However, stochastically the points coincide, due to the central limit theorem.}
6.2 Advanced methods

Improving predictions using other sensory information could improve the accuracy of usage pattern detection. This is feasible provided that significant coefficients can be isolated from correlating these data. This requires a significant amount of usable samples. A criterion the ramifications of which are further exacerbated by the influence of unaccounted variables. Time of usage profiles are influenced for instance by seasonal and weather fluctuations. Including these variables increases the dimensionality of the prediction problem. However, modulating predictions using these variables will increase accuracy if the prediction model is specified accordingly. Correlating patterns on a per-user basis can reveal patterns that can improve predictions. If motion or temperature based detection sensors are included, human presence can be used to specify the need for use of light or heating. Furthermore, correlations between usage of applications can be explored. Most importantly, inconsistencies in data are prevented when external influences are accounted for in the prediction of behaviour as well as the prediction of yield from alternative energy sources.
Chapter 7

Implementation

In this section, we describe how we have translated our theoretical ideas about data analysis into practical code. This can be divided into two principal subsections: the prediction of future behaviour based on historical data, and decision-making based on those predictions.

7.1 Prediction

Our prediction algorithms attempt to forecast the power production or consumption of a given node for a certain time into the future – say, 24 hours. They produce a vector in which each element corresponds to some timestamp in the future:

\[ F = [F_1, F_2, F_3, \ldots] \]  \hspace{1cm} (7.1)

If the data is logged with a sampling period of \( t_s \) seconds, then \( F_1 \) corresponds to the predicted production or consumption \( t_s \) seconds from now, \( F_2 \) to the predicted value \( 2t_s \) seconds from now, and so on. (By convention, a positive value of \( F_\tau \) corresponds to power consumption, a negative value to power production.)

To arrive at these predictions, we make use of two axioms:

- A node’s behaviour depends on the time of day and the time of year. Many appliances are only switched on at specific times of day and rarely or not at all at other hours; and of course solar panels do not produce a lot of
power in the middle of the night. The yearly cycle, with its more or less regular variation in sunlight hours and temperature, also plays a part: on an afternoon in December the lights will be switched on at an earlier hour than on an afternoon in October, and throughout the summer months few users will heat their homes at all.

- A node’s behaviour depends on its recent history. Especially on the load side, it is very useful to be able to detect so-called load signatures. A washing machine, for example, might always go through a certain fixed pattern from the moment it is switched on, regardless of the time of day at which this happens; thus, by looking at the recent values of the machine’s power consumption, an algorithm might recognise part of this pattern and predict that the upcoming values will also follow the pattern.

We will now describe the code we have written based on these axioms.

### 7.1.1 Prediction by time of day

These functions assume that the historical data is timestamped: for each recorded value, a time and a date are specified. Moreover, it assumes a fixed sampling period of $t_s$ seconds, so that there are always $\frac{3600}{t_s}$ samples in an hour and $\frac{86400}{t_s}$ in a day. (If the latter is not true, it can be easily realised by filling up the empty timestamps with the previous recorded value – as long as the data are timestamped.)

Given this set of timestamped data, predicting by time of day is very simple – just find the mean value for each timestamp:

- Start with $\Sigma_k = 0$ and $N_k = 0$ for all possible timestamps $k$.
- Loop over the historical data. Whenever you encounter a value timestamped with $j$, increment $\Sigma_j$ by that value and $N_j$ by 1.
- When finished, divide $\Sigma_k$ by $N_k$ for each $k$.

(Note that we use timestamp to refer specifically to the time of day at which a value was recorded; for the date, we shall use the term datestamp.)

Now, given these averages and the current time $t$, we can fill the forecast vector $F$:

$$F_\tau = \frac{\Sigma_{t+\tau}}{N_{t+\tau}}$$  \hfill (7.2)
For example, if \( t_s = 60 \) (one sample a minute) and the current time is 15:00, then \( F_{120} \) will be the average value for 17:00. Of course, the addition \( t + \tau \) needs to wrap around – more formally, it needs to be performed modulo \( \frac{86400}{t_s} \) – so that if the current time is 23:30, \( F_{60} \) will be the average value for 00:30.

Time-of-year prediction is a slightly modified version of time-of-day prediction: it takes the datetimestamp as well as the timestamp into account. Replace the mean value for each timestamp above with the mean value for each timestamp-datestamp pair, and the rest of the description remains valid.

### 7.1.2 Prediction by Markov chain (first-order)

This function assumes that the power at a given node is quantised to a finite and manageable number of possible values. (This is trivial to realise, since the data from the nodes is already received in digital format anyway. Reduction of the number of possible values by a factor of \( 2^n \) can be achieved by throwing away the last \( n \) bits of each received value – a sort of downsampling in signal value rather than time.)

Now, given this finite set of values that the power can assume, the function constructs a simple empirical Markov model. That is to say, it counts how often a given value is followed by a given other value in the existing record of the node’s behaviour, then normalises those numbers to arrive at a matrix of transition probabilities. More concretely:

- Start with an \( n \times n \) matrix of zeroes, where \( n \) is the number of possible values.
- Loop over the historical data. Whenever the value \( i \) is followed by the value \( j \), increment the \( j \)th element of the \( i \)th row by 1.
- Finally, divide each element by the sum of the row it appears in (i.e. by the total number of times the value \( i \) has occurred).

The end result is a matrix of transition probabilities. If, for example, there are 1000 instances of the value 6 throughout the dataset, 400 of which are followed by the value 7, then \( P_{6,7} = 0.4 \).

This matrix can now be used for prediction. If the most recent observed value \( x_t \) is equal to \( a \), then the PMF (probability mass function) for \( x_{t+1} \) is simply the \( a \)th row of the transition matrix. From this PMF, an expected value can be calculated in the usual way; this is our forecast for \( x_{t+1} \).
\[ F_1 = E[x_{t+1}] \quad (7.3) \]

For the value of \( x_{t+2} \), we have to proceed a little differently, as we no longer have a single fixed value to depart from. Of course, we could simply assume \( x_{t+1} \) will take on its expected value, and use the \( E[x_{t+1}] \) row of the transition matrix as our PMF for \( x_{t+2} \), but this is crude and throws away a lot of information. A more elegant solution is to remember the PMF for \( x_{t+1} \), and use a probability tree: multiply \( P[x_{t+2} = c | x_{t+1} = b] \) with \( P[x_{t+1} = b] \) for all values of \( b \) and \( c \), and then sum the probabilities of all paths that lead to the same \( c \). In equation form:

\[ P[x_{t+2} = c] = \sum_b P[x_{t+2} = c | x_{t+1} = b] \cdot P[x_{t+1} = b] \quad (7.4) \]

We now have a PMF for \( x_{t+2} \), from which we can again calculate an expected value, which we append to our forecast vector \( F \).

At first sight this seems to lead to an exponential growth of the number of calculations: there are \( n \) possible paths from \( x_t \) to \( x_{t+1} \), \( n^2 \) paths from \( x_t \) to \( x_{t+2} \), \( n^3 \) paths from \( x_t \) to \( x_{t+3} \)... In practice, however, we only need to know the PMF for \( x_{t+\tau} \) and the transition matrix to calculate the PMF for \( x_{t+\tau+1} \); how we arrived at the PMF for \( x_{t+\tau} \) is irrelevant and will not influence the PMF for \( x_{t+\tau+1} \) in any way. Therefore, equation 7.4 can be used equally well to calculate the PMF for \( x_{t+10000} \) from that for \( x_{t+9999} \), and the number of paths to investigate is constant at \( n^\tau \).

### 7.1.3 Prediction by load signature (higher-order)

Unlike the first-order prediction function, which only looks at the most recent value, this function takes the \( L \) most recent samples into account. These are stored in a vector \( H \):

\[ H = [x_{t-L+1} \quad \cdots \quad x_t] \quad (7.5) \]

The function loops over the historical data, checking if the sequence \( H \) has occurred before. If it finds a match with \( H \) (within certain bounds of tolerance), it stores the next \( M \) samples in a vector \( O_k \):

\[ [x_{\nu-L+1} \quad \cdots \quad x_\nu] \approx H \Rightarrow O_k = [x_{\nu+1} \quad \cdots \quad x_{\nu+M}] \quad (7.6) \]
When this loop is finished, the various observed sequences $O_k$ are compared. The predicted future values for $x_t + 1$ through $x_t + M$ are now simply the averages of the corresponding elements of $O_k$:

$$F_{\tau} = \frac{\sum k O_{k,\tau}}{K}$$

(7.7)

Here $K$ is the total number of matches with $H$ that have been found in the historical data, and hence also the total number of observations $O_k$.

### 7.1.4 Combining different predictions

The final forecast is a weighted sum of the three predictions (time-of-day, first-order, higher-order) described above:

$$F_{fin,\tau} = \frac{w_{tod,\tau} \cdot F_{tod,\tau} + w_{fo,\tau} \cdot F_{fo,\tau} + w_{ho,\tau} \cdot F_{ho,\tau}}{w_{tod,\tau} + w_{fo,\tau} + w_{ho,\tau}}$$

(7.8)

How are the weighing vectors $w_{tod}$, $w_{fo}$ and $w_{ho}$ generated? Recall that the elements of $F_{tod}$ and $F_{ho}$ are calculated as averages of certain sets (see equation 7.2 and 7.7, respectively). Here we can use the variance of those sets as a measure for the strength of the prediction (if the historical values for e.g. 14:00 are all tightly clustered around a certain value, then the time-of-day-based prediction for 14:00 will be stronger than if the values for 14:00 are widely scattered). Hence:

$$w_{tod,\tau} = \frac{1}{\text{Var}[X_{t+\tau}]}$$

(7.9)

where $t$ is the current time and $X_{t+\tau}$ is the set of all historical values for the timestamp $t + \tau$; and:

$$w_{ho,\tau} = \frac{1}{\text{Var}[D_\tau]}$$

(7.10)

where $D_\tau$ is the set of $\tau$th elements of all vectors $O_k$ as described in subsection 7.1.3.

By contrast, the elements of $F_{fo}$ are expected values based on a PMF (see equation 7.3). Here, we use the highest probability that occurs in the PMF to
indicate the strength of the prediction: if there is a very high chance of one specific value occurring, the prediction is more certain.

\[ w_{j_0, \tau} = \max\{P[x_{t+\tau} = c]\} \quad (7.11) \]

## 7.2 Decision-making

The action options for the data analysis unit are limited, as the MCU already enforces static balance of power. To improve upon the MCU’s static functionality, there are two dynamic actions available:

- Start charging up a battery
- Switch on a load that has been specified to run at some point in the next 24 hours (see section 2.1.3)

We will now explain how the system uses its own predictions to make these decisions.

Given that we have a forecast vector \( F \) for each node, we can generate a forecast for the total net power production or consumption in the system – let us call it \( G \) – by summing them all:

\[ G_\tau = \sum_{F} F_\tau \quad (7.12) \]

Charging up a battery is a measure to be taken when there is enough power available now, but a shortage is foreseen in the near future. To determine whether this is the case, the elements of \( G \) are summed in order, until the sum drops below zero. The element of \( G \) at which this happens is marked as \( \tau_0 \). Concretely, the significance of \( \tau_0 \) is: the analysis module predicts that the total energy consumed in the system over the following \( \tau_0 \cdot t_s \) seconds will be larger than the total energy produced during the same time. In other words: given enough battery capacity, the system can hold out for another \( \tau_0 \cdot t_s \) seconds. If \( \tau_0 \) drops below a certain threshold (\( \tau_0 < \tau_1 \)), the system will instruct the MCU to start charging a battery.

For scheduling decisions – the second dynamic action mentioned above – the approach is slightly different: \( G \) is broken up into time segments (say, 1 or 2 hours each) and for each segment, the average value is calculated. The system
then chooses the segment with the highest value to schedule the load. Take, for example, a dishwasher that has been configured to run at some point in the next 24 hours. If the system predicts there will be more power available in four hours than there is now, the dishwasher's operation will be delayed until then; if there is more power available now than there will be at later times, then this is the best time to schedule the load, and the MCU will be instructed to switch on the dishwasher.
Chapter 8

Evaluation

In this chapter, we will describe how we tested the prediction algorithms from chapter 7, and what the results were.

8.1 Input data and testing process

We have used the REDD low frequency dataset, provided by the authors of [4], to test prediction algorithms. This set contains apparent power consumption data for appliances in six different households over several weeks. Originally sampled at slightly below 1 Hz, we downsampled it to one sample per 10 minutes in order to make calculations less unwieldy.

For each appliance, two random timestamp-datestamp pairs were chosen. Starting from both of these times, the prediction algorithms described in chapter 7 were run: the Markov prediction 5 hours (30 samples) ahead, the time-of-day and signature predictions 24 hours (144 samples) ahead. The generated predictions were then compared to the actual data for the corresponding timespan.

8.2 Prediction by time of day

The accuracy of time-of-day-based prediction (see section 7.1.1) varied. Many appliances were simply switched off — power consumption 0 — during the 24-hour timespan under consideration. Usually, the prediction also stayed very close to zero in these cases (indicating that the appliance was never or rarely switched
on during the observed weeks at all), although there were exceptions.

This algorithm achieved its best results for lighting devices. For these appliances, the prediction often tracked the *timing* of consumption peaks fairly accurately, but significantly underestimated their *magnitude*. Given the averaging functionality of the time-of-day prediction algorithm, this suggests that lighting devices were not switched on every day, but on those days they were switched on it happened at more or less the same hours. Figures 8.1 through 8.5 illustrate this behaviour.

*(Note: For all plots in this chapter, the Y-axis is labeled in watts, and the X-axis in samples ahead, where 1 sample = 10 minutes.)*

![Figure 8.1: Time-of-day-based prediction for lights from the REDD dataset (house 1, socket 17)](image)
Figure 8.2: Time-of-day-based prediction for lights from the REDD dataset (house 2, socket 4)

Figure 8.3: Time-of-day-based prediction for lights from the REDD dataset (house 3, socket 17)
Figure 8.4: Time-of-day-based prediction for lights from the REDD dataset (house 4, socket 3)

Figure 8.5: Time-of-day-based prediction for lights from the REDD dataset (house 4, socket 13)
For other types of appliances the prediction occasionally showed similar behaviour, as seen in figures 8.6 and 8.7.

Figure 8.6: Time-of-day-based prediction for kitchen outlets from the REDD dataset (house 4, socket 5)
For appliances whose consumption pattern showed high-amplitude periodic variations (such as refrigerators), the algorithm showed an interesting "smoothing" function: it remained relatively constant at some level between the peaks and valleys of the cycle. This is understandable; if the periodicity of the consumption pattern is out of sync with the 24-hour cycle, the peaks will occur at different times of day, and thus their contributions are "smeared out" over the averages for different timestamps. This effect can be seen in figures 8.8 and 8.9.
Figure 8.8: Time-of-day-based prediction for kitchen outlets from the REDD dataset (house 1, socket 7)

Figure 8.9: Time-of-day-based prediction for a refrigerator from the REDD dataset (house 3, socket 7)
8.3 Prediction by Markov chain (first-order)

With regards to the Markov-chain-based prediction algorithm from section 7.1.2, we can be brief: it does not have any predictive value at all. We deliberately chose to run it for a shorter timespan than the other algorithms (5 hours instead of 24) because we felt it would not be able to make meaningful predictions further ahead – but even that was too optimistic.

The power usage patterns predicted by this algorithm were usually either reasonably smooth linear curves, or curves that rapidly ascended or descended to a certain "equilibrium point" and stayed there – presumably, in the latter case, the predicted value reached a level with very high transition probabilities to itself or to nearby levels. Any relation to the actual data was absent – except in very rare "lucky" cases such as shown in figure 8.10:

![Figure 8.10: Markov-chain-based prediction for a refrigerator from the REDD dataset (house 6, socket 8)](image)

Figures 8.11 through 8.14 are more representative of the algorithm’s behaviour.
Figure 8.11: Markov-chain-based prediction for a refrigerator from the REDD dataset (house 1, socket 5)

Figure 8.12: Markov-chain-based prediction for kitchen outlets from the REDD dataset (house 1, socket 7)
Figure 8.13: Markov-chain-based prediction for lights from the REDD dataset (house 1, socket 9)

Figure 8.14: Markov-chain-based prediction for a furnace from the REDD dataset (house 4, socket 4)
8.4 Prediction by load signature (higher-order)

Given its averaging nature, the signature-based prediction algorithm from section 7.1.3 often showed similar behaviour to the time-of-day algorithm – getting the timing of peaks more or less right, but the magnitude wrong, as seen in figures 8.15 and 8.16.

Figure 8.15: Signature-based prediction for lights from the REDD dataset (house 1, socket 9)
In one case, an interesting reversal of this behaviour occurred when the algorithm perfectly predicted the load cycle of a refrigerator – except delayed by about 12 hours, as seen in figure 8.17:
Signature-based prediction really shone, however, when used on appliances with highly cyclic consumption patterns. By tracking at which point in the cycle the appliance was at the moment of prediction, the algorithm was, in several cases, able to predict 24 hours ahead with perfect accuracy – see figures 8.18 through 8.21.

Figure 8.17: Signature-based prediction for a refrigerator from the REDD dataset (house 5, socket 18)
Figure 8.18: Signature-based prediction for a refrigerator from the REDD dataset (house 1, socket 5)

Figure 8.19: Signature-based prediction for a furnace from the REDD dataset (house 4, socket 4)
Figure 8.20: Signature-based prediction for kitchen outlets from the REDD dataset (house 4, socket 14)

Figure 8.21: Signature-based prediction for a refrigerator from the REDD dataset (house 6, socket 8)
8.5 Combined prediction

As described in section 7.1.4, the total prediction is a weighted sum of the different algorithms’ outcomes, the idea being to reach a more accurate prediction by combining the influence of different factors and weighing stronger predictions more heavily.

This was modestly effective. There were few cases where the combined prediction was noticeably close to reality when neither of the individual components were – in other words, where “two wrongs made a right”. Figure 8.22 shows one of those cases. The appliance was switched off throughout the 24-hour timespan. The time-of-day and signature algorithms both predicted a peak, but at different times, and both times the non-peaking algorithm was assigned higher weight. Thus, by combining the two, the final prediction stayed relatively close to zero (and hence to reality) throughout.

Figure 8.22: Combined predictions for an unknown type of appliance from the REDD dataset (house 3, socket 12)

Figure 8.23 shows the average (absolute) prediction error for each of the prediction methods. Note the simultaneous peaks in the different graphs; these are likely caused by very high peaks in specific appliances’ consumption data (e.g. that of a microwave oven) that were missed by all algorithms and thus "pull up" the average error at the time for which they occurred.
That being said, it is clear that the weighted sum of all predictions has a lower average error than any individual prediction method throughout. This shows that using a weighted sum of different predictions to arrive at a more accurate total prediction, with the weighing factors as described in section 7.1.4, is effective – if only modestly so.

![Figure 8.23: Average error for different prediction algorithms](image)

### 8.6 Conclusion

We can summarise our conclusions with regard to the different prediction algorithms’ effectiveness as follows:

- **Time-of-day-based prediction** works best for lighting devices, predicting with decent accuracy when consumption peaks will occur, but usually underestimating their magnitude. For appliances with high-amplitude cyclic behaviour, the prediction is often a "smeared-out" version of the real signal – which will still lead to an accurate outcome when the prediction is used to calculate the average power consumption over a longer period.

- **Signature-based prediction** works excellently on appliances with highly regular, cyclic behaviour, such as refrigerators. For other appliances, it often mimics the "right time, wrong magnitude" behaviour of the time-of-day algorithm.
• First-order Markov-chain-based prediction is useless.

• The weighted sum of all predictions is on average more accurate than any prediction by itself, but this effect is not spectacularly large.
Chapter 9

Discussion

In section 7.2, two parameters were derived from the grand total forecast $G$: a time-until-equilibrium $\tau$, and an average power value for each of a number of time segments. Both involve integrating $G$ over a certain intervals. These predictions are sensitive to:

- the length $\tau$ of the sampling interval
- the accuracy of the averaging of power in the interval
- the quality of the predicted load signature

The sampling interval determines the time interval over which power is averaged. This results in an energy block consisting of the predicted average power $E_{P_{avg}}$ for the interval. Elongating this interval will on one hand help temper $E_{P_{avg}}$. Contrastingly, elongated intervals will smooth out the effect of surges and switching behaviour. This entails an inability to account for peaks in demand, and therefore accounting for these effects in maintaining the power balance. For mitigation of these limitations refer to the quality of predicted load signatures below.

The accuracy of the averaging of power over an interval is dependent on the resolution of the sampling of the power signature. Both the ability to extract power signatures as well as the average power dissipation over a certain period is improved with an increased sampling interval. The quality of the predicted load signature for an appliance determines the specification of the predicted behaviour of a load. This behaviour according to the load signature determines the future behaviour of a load based on the current state. Therefore any error in the
nature of this signature will propagate through into $E_{P_{avg}}$ accordingly. Furthermore, it bears noting that a higher sampling rate reduces the misidentification of signatures due to variance in the offset between signature and sampling.

The expected average load $E_{P_{avg}}$ is determined by

$$E_{P_{avg}}^{(j)} = \sum_{i \leq j} v_i E_{v_i} [\tau]$$  \hspace{1cm} (9.1)

The sum of expected power states $v_i$ and the expected durations $E_{v_i} [\tau]$ for these states. The expected duration is the expected value for the duration of state level $v_i$. Errors in the duration arise from the resolution due to the sampling frequency for appliance loads. Furthermore, if state durations do not have consistent usage patterns, the variance on the expected duration propagates as uncertainty in $E_{s_i}$, such that $E_{s_i} = \tau_{s_i} \pm \sigma_{e_{s_i}}$. These uncertainties propagate through into $E_{P_{avg}}$ according to the sample variance

$$\varsigma_{P_{avg}} = \frac{\sum(E_P - |E_D|)^2}{N - 1}, \text{covar}(E_i, E_j) = 0 \forall i \neq j. \hspace{1cm} (9.2)$$

Note that $|s_{n_i}|$ denotes the expected power dissipation and is represented here independent of the uncertainty related to the value. Furthermore, since loads and their signatures are independent, the covariance is zero. Clearly, the propagation is relative to the power level as well as the variance on the expected state duration for each signature. Errors in the predicted power levels result in misrepresentation of states, hence the absolute error in the predicted average power is

$$|e_{P_{avg}}| \leq |e_{v_{n_1}} E_{v_{n_1}}| + \cdots + |e_{v_{n_K}} E_{v_{n_K}}| \hspace{1cm} (9.3)$$

Which may be significant when evaluated as a percentage of $E_{P_{avg}}$ due to the possibly relatively large energy contribution of some element. Thus, the percentage-wise error requires the weighing of the individual errors with respect to the load they modulate, as well as with respect to the contribution of this load to the composite expected power requirement. However, it may be assumed that if such a situation occurs the consequences are marginal, since this load is in this case the critical element in the system.

The accuracy of the prediction depends on the stationary transition probability matrices and the variance on the expected state transition wait time. For example, consider the first order Markov chain consisting of the initial state $v_0$, and the transition matrix
\[ \pi_{v_0} = \begin{bmatrix} 0 & 9/10 & 1/10 \\ 9/10 & 0 & 9/10 \\ 1/10 & 9/10 & 0 \end{bmatrix} \] (9.4)

In this example the transition has \( \pi_{ij} > 0 \) for state transitions \( \pi_{0 \rightarrow 1} = 9/10 \) as well as \( \pi_{0 \rightarrow 2} = 1/10 \). Disregarding the uncertainties on these percentages due to the sample size from which these were obtained for a moment, the percentages themselves suggest there is a one in ten probability that assuming \( \pi_{0 \rightarrow 1} \) will fail. Equally, the variance on the expected wait time relates a time interval to a probability of a state change occurring in that interval. Trivially, the probability of a state change occurring in an interval \( \pm \sigma \) around \( \mu_{\text{state change}} \) is roughly 68%.

The size of the interval is a function of the standard deviation \( \sigma \). As shown in 9.1 two instances of a dishwasher power signature do not match exactly. The signatures are comparable if some variance is permitted on state durations.

![Figure 9.1: two instances of measured power signatures from a washer-dryer combination sampled at 1 Hz. Data provided by [4].](image)

Averaging load signatures over 5 samples reduces the influence of higher order effects on the signal. Such a filtering reveals a load signature of more or less identifiable and constant states, and of comparable character. As shown in 9.2 where five instances of the power signature of a washer-dryer combination taken from [4] are shown together with the signature averaged over five samples and upsampled back to 1 Hz. From this example it is hypothesized that some extra identifier is required to ensure proper detection of a state change, given that the averaging sometimes normalizes out the notches indicating a change in operation.

In the figure a green dashed line is drawn at the 300W point. It is moderately effective at detecting notches in the signature. This approach turns a notch in the signature into an extra state. This threshold was established by visual
analysis of the pattern. It may be possible to automate this process using some regression method. However, a universally applicable algorithm is complicated by the admission of various load types.

Furthermore, appropriate scaling methods must be implemented if no training data is available. As a preliminary solution the first order derivatives $v_i - v_{i+1}$ are used, again taken above some threshold value (in this instance $|\delta_{i+1}| > 300$). These values are also used as the state change instances to analyse the efficiency of the proposed parameters for constructing load signature patterns. Interpretation of large derivatives as possible state changes provides opportunities to increase robustness by verifying the likelihood of the resulting level according to the Markov model.

Figure 9.2: Five instances of a measured power signature from a washer-dryer combination sampled at 1 Hz. The original signal is shown in red. The sample was truncated at the first sample > 400 (power in Watt). In black is the signal reconstructed as the average over five samples, Triangles indicate where the first order derivative of the original sampling exceeds a $\Delta > 300$ threshold. Data provided by [4].

Although methods for automated appropriation of useful values for the thresholds described above have not been implemented, the handles described above have been applied to a random length of sample data of the same appliance. The string, and highlighted samples from it are given in figure 9.3. The graphs reveal false positives of large derivatives that do not correspond with significant state changes, indicated by triangles in the figure. This suggests improvement may be possible if the derivatives are related to large derivatives in the averaged set. As indicated in chapter 6, manipulating the averaging window will influence both the variance in a supposedly constant level, as well as what would be appropriate threshold values.
Figure 9.3: simulation of detection algorithm for a load signature based on averaging and state transitions
Scrutiny of the proposed approach is conditional on proper implementation for the scaling of threshold values to adapt to various load signatures. The methods detailed here do not include the feasibility of construction a meaningful stationary distribution on which a Markov model could be constructed. Theory concerning Markov models [3][5], and their application in pattern detection [6] and load detection [1][7] are well established, even in non-intrusive monitoring applications. However, further testing is required before meaningful conclusions can be drawn about the applicability of the proposed methods in environments where various load signature patterns can occur.
Chapter 10

Suggestions for future work

The current system is dependent on a large set of data to obtain meaningful results. It may be possible to devise algorithms that are able to make predictions based on less information by exploring optimization strategies [8] and their applicability. It may also be postulated that if the full potential of measured variables available to the system is available at all times, the decision making process could obtain comparable results using only rough bits of trend information. These alternative approaches may yield useful improvements to the existing methods.

The current system does not use training sets to generate prediction models. Assuming some continuity and recognisability of the loads connected to the system, the availability of a training set would improve the ability to recognize anticipated signatures. The use of nearest-neighbour methods [7], under the assumption that a signal is comparable to known sequences might well improve results significantly. However, without a priori knowledge about the device connected, further methods are required to match trained signatures to connected loads, in some ways analogous to non-intrusive load monitoring methods [1].

Given the assumption of load signature patterns with limited states devising training sequences for types of load signatures may improve results significantly. Feasibility of training sequences for parametric recognition of load signature types is suggested for future work.
Literature
Bibliography


