Imperial College London
Department of Computing

# Inferring Appliance-level Energy Consumption from Whole-House Electricity Meter Readings Using Edge Detection and Appliance Models

by

Simon J. Leigh

Submitted in partial fulfilment of the requirements for the
MSc Degree in Computing Science of Imperial College London

September 2014

**Abstract**

Energy conservation is increasingly a priority for individuals and governments alike. The reluctance of European governments to sanction new nuclear power plants in the face of public opposition, and the gradual 'mothballing' of older plants has resulted in an increased reliance on fossil fuels. Renewable sources of energy are beginning to make larger contributions to the total supply, but all suffer the problem that they are not available on-demand. In both the developed and developing world, consumers demand more energy as their use of household appliances increases. In the UK, up to 30% of electrical energy produced is consumed by the residential sector. The UK has legislated to have smart-meters installed at every home within the next decade.

The wide-spread deployment of smart meters will allow consumers more insight into their electricity consumption than ever before, and energy providers and the Grid more information with which to manage demand. Whilst providing consumers with real-time information on their total power consumption is of benefit, of even greater value is actually providing information on which appliances consume the most energy. This would change the communication with consumers from a monthly bill, to potentially real-time feedback of what each appliance is costing them. Empowered with this information, people are in a better position to decide where they are able to make savings in their energy consumption.

Non-intrusive load monitoring has been researched for almost 30 years. Its goal is to extract appliance-level information from aggregate data of whole-house consumption. Despite ongoing research, there is no commonly agreed on approach that has provided a solution ready for consumer use on a large scale. There are also no commonly agreed upon 'benchmark' algorithms. Software which enables researchers to easily compare their disaggregation accuracy is only just becoming available.

This research implements a disaggregation system based upon detecting steady-state power changes, a previously successful method. Appliance-models (in this case, built from ground-truth measurements or specified by the user) are then used to identify the most likely device to have been responsible for each steady-state of power.

Occupancy and temperature data were also collected alongside power data. Whilst not implemented into the disaggregation algorithm, an analysis has been undertaken which identifies both occupancy and temperature data as a potentially useful in future disaggregation approaches building on the techniques used.

The results of applying the system to data collected at the author's home show success in disaggregating several different classes of appliance, including the refrigerator, dishwasher and electric hob. However, the fraction of energy correctly assigned to devices over a month's worth of data was at best 40%. Changes in the characteristics of household appliances since the approach was first used successfully (1992) are identified as undermining certain aspects of the approach and informing future iterations of the system.

Dedicated to my grandparents.

# Acknowledgements

I would like to thank my parents for their constant and unfailing support (both financial and moral) without which I would likely have had neither the courage nor means to return to full-time study.

I would also like to thank Jack Kelly for initiating my interest in this area, for his support during this project (and whatPlant!), for writing much of NILMTK (which made my life considerably easier), for designing the metering apparatus and software I used, and for giving me the chance to attend NILM Workshop 2014 which was both extremely informative and enjoyable.

Dr William Knottenbelt has my thanks for his incredible enthusiasm, support, approachability, and for his energy and clarity in teaching.

Tom Green: sleeping is giving in, no matter what the time is! Thank you being such a friend and for suggesting that I study at Imperial.

Finally, I must thank Helen. She has had endless patience with me over the past year and been a source of great encouragement from the beginning. Thank you for your support, love and affection.

# Contents

# Chapter 1

# Introduction

Faced with the prospect of significant human-influenced climate change, governments and multi-national organisations are attempting to motivate individuals to reduce their energy consumption and wastage. Most will be familiar with the calls to reduce the temperature of one's home thermostat and to switch off the lights when leaving a room. Beyond these basic measures, how can individuals be made more aware of areas where they might make gains in their energy efficiency? It is important to note that the benefit is not only to consumers: carbon emissions are reduced as electricity consumption declines and less stress is placed on the electrical grid at peak hours.

Studies have shown that providing consumers with appliance-level information can significantly increase the reductions they are able to make in overall consumption [8]. If you knew that by replacing your ageing electrical equipment with newer, more efficient models would pay for itself in electricity costs within 18-months, you might be much more likely to take action. How can we provide this data to individuals without placing 'invasive' load monitors on their household appliances?

Non-Intrusive Load Monitoring (NILM) aims to provide appliance-level information from 'whole home' electricity consumption data. The aim of this project is to implement software that can extract appliance level consumption data from the aggregate data of a household.

## 1.1 Background

### 1.1.1 Electricity Supply in the UK

Electricity generation in the UK fell by 1.0 per cent from 367 TWh in 2011 to 364 TWh in 2012 (terrawatt hours, $10^{12}$ Watts $\times$ 3600 seconds: a measure of energy). Gas's share of generation fell from 40 per cent to 28 per cent over the same period, whilst coal's share rose from 30 to 39 per cent, as a result of the increasing price of gas relative to coal. Domestic consumption of electricity increased by 2.8 per cent over the period (though this is attributed to a colder than usual final quarter) [30].

The UK is a net importer of electricity: our generation capacity does not meet the aggregate demand. In 2012, the UK had net electricity imports of 3.2 per cent. The highest level since 2000 [30]. Electricity is imported (or exported) via large undersea 'interconnectors' to continental Europe and Northern Ireland.

Domestic consumption made up 30 per cent of UK electricity demand in 2012: The largest individual sector contributing to the aggregate demand. [30].

Figure 1.3a shows the large contribution of coal to the UK's electricity supply. The UK has 17 coal power stations, the proportion of overall electricity generated by them can rise to 50 per cent at peak periods during winter. The carbon emissions associated with electricity production from coal are around twice that of gas [22].

In addition to a significant dependence on coal produced electricity, the UK faces a supply issue. Ofgem state in their 2014 Electricity Capacity Assessment Report [48]:

> 'The supply outlook has continued to deteriorate since last years assessment. Generators have withdrawn or announced their intention to withdraw around 3GW of plant in the next two years. A further 2GW of plant was already scheduled to close before the end of 2015, due to emission standards and plant reaching the end of their lifetime.'
>
> 'National Grid projects that the supply side outlook will deteriorate until the mid-decade in all FES [Future Energy Scenarios] scenarios. It assumes that around 5GW of conventional plant will shut down permanently in the next two winters and an additional 1GW of gas plant will mothball in the same period. The supply outlook is the same for all FES until the middle of the decade. This is a worse supply outlook than last years FES.'

National Grid (the organisation responsible for the maintenance of the UK's gas and electricity supply infrastructure) has published the 'UK Future Energy Scenarios 2014' (FES) report which illustrates that there are three factors affecting the UK's future energy security. Supply, affordability and sustainability. All three are impacted by political and economic factors, and thus there can be no certainty of a move toward more sustainable energy supply [44].

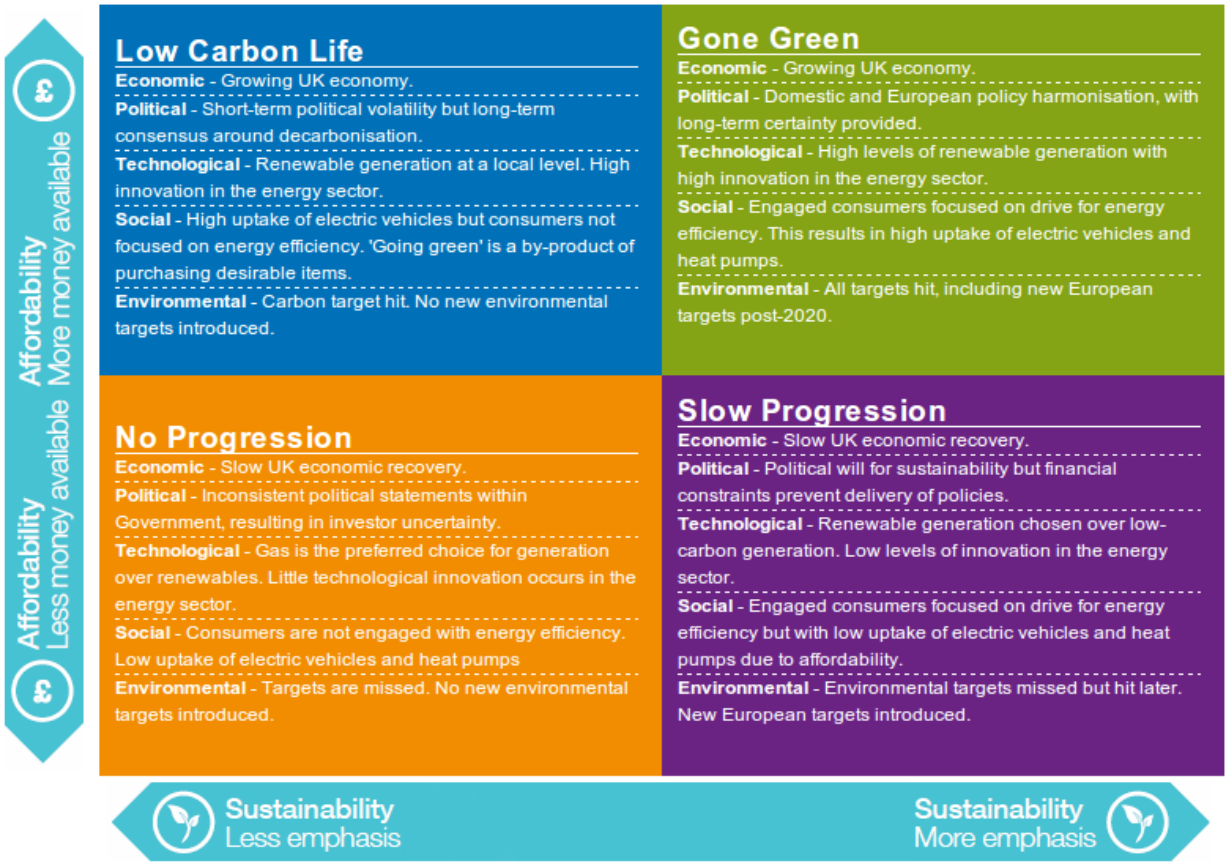



**Figure 1.1:** 'National Grid 2014 Future Energy Scenarios' (National Grid, 2014:p.5)

Political will to reduce carbon emissions varies considerably between the major political parties in the UK, this is in evidence in a comparison of the UK Labour and Conservative parties' 2010 election manifestos [16], [39]. The impact of the affordability of any measures to reduce carbon emissions and the national approach to sustainability is illustrated in Figure 1.1. This Figure demonstrates four broad categories of outcome dependent on capital availability and degree of emphasis on sustainability applied at a National level.

In spite of the uncertainty of outcome, in FES, National Grid find that the UK Smart Meter roll out is likely to complete by 2020 in the best case and in the worst case (with minimum economic growth and low emphasis on sustainability) by 2032 [44].

The introduction of Smart Meters is likely to be combined with time-of-use-tariffs (TOUTs) and could thus facilitate the smoothing of intra-day demands (known as demand management) to reduce the relative difference between demand peak and demand minimum. It may also enable the enhanced use of intermittent renewable generation such as wind or solar [44].

In their Electricity Capacity Assessment Report 2014, Ofgem release their estimates for the 'Loss of Load Expectation', they define it as follows [48]:

> 'This is the average number of hours in a year where we expect National Grid may need to take action that goes beyond normal market operations. Importantly, this still does not represent the likelihood of customer disconnections. Controlled disconnections of customers - typically industrial and commercial sites before households - would only take place if a large deficit were to occur.'

Ofgem find that:

> "Under National Grids scenarios, loss of load expectation increases from less than 1 hour per year in 2014/15 to a maximum of around 3 to 5 hours per year in 2015/16. Our analysis shows a larger range of risks, from as much as 9 hours per year for a higher demand or lower supply, down to close to zero if, for example, there were full imports from mainland Europe."

Figure 1.2 shows the range of LOLE for the period 2014 to 2019. It can be seen that in the short to medium term there is an expectation of a few hours loss-of-load per year, with a significant degree of uncertainty. As these figures depend on full imports from Europe, the political climate and stability of the region may have a significant influence on the actual loss-of-load figure.

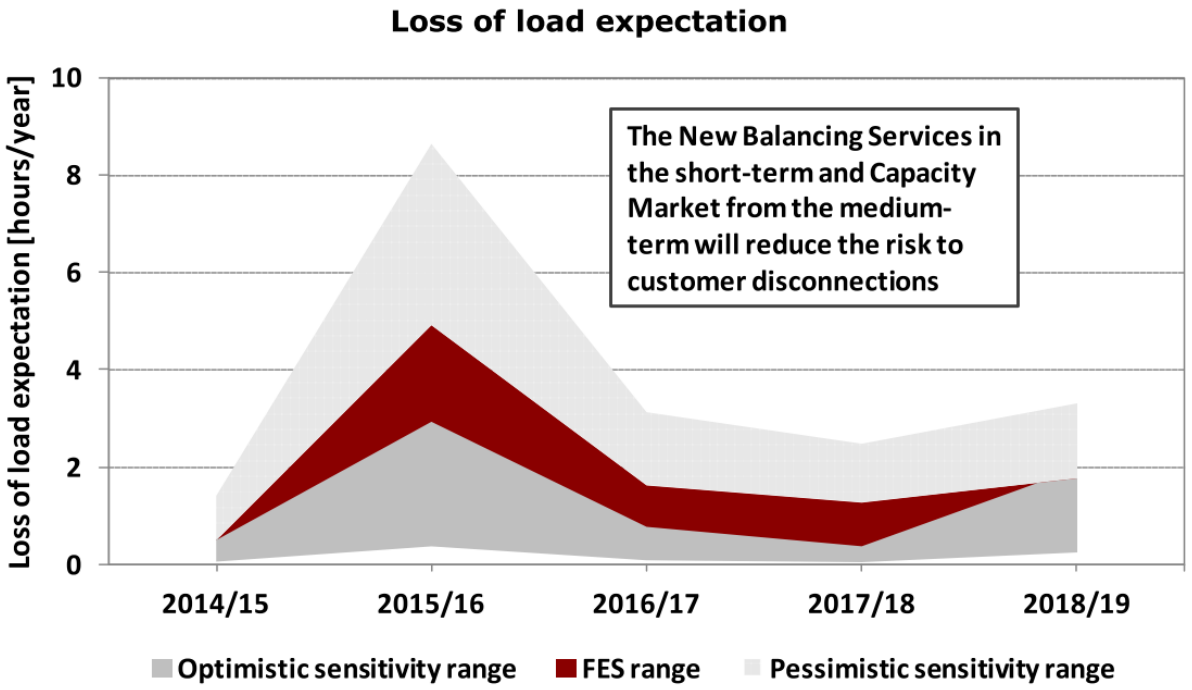
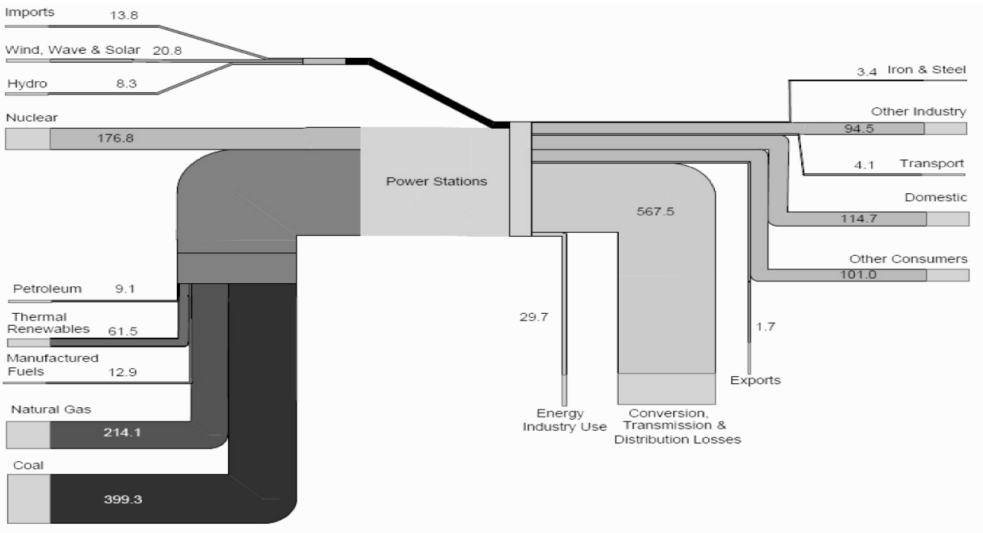
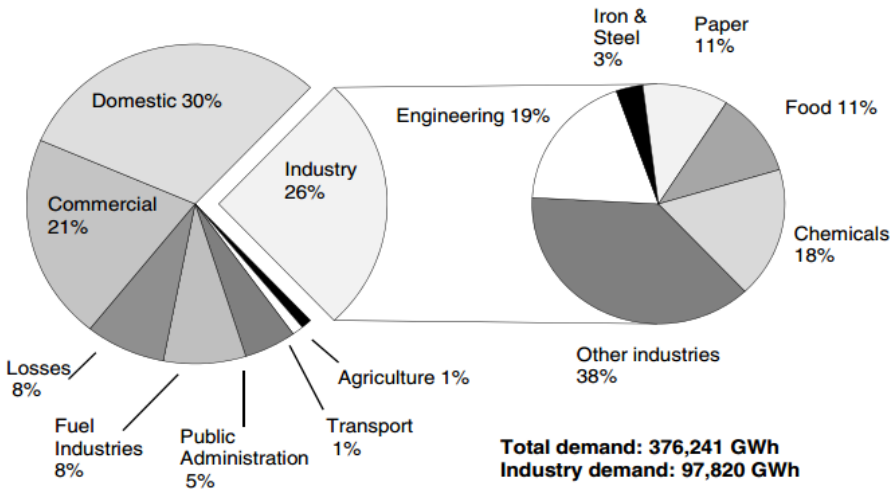



**Figure 1.2:** 'Loss of Load Expectation' (Ofgem, 2014:p.6)




**(a)** 'UK Electricity Flow Chart 2012 (TWh)'




**(b)** 'UK Electricity demand by sector 2012'

**Figure 1.3:** UK Electricity Data (UK Department of Energy & Climate Change, DUKES, 2013:p.112-115)

### 1.1.2 Carbon Emissions

Globally increasing demands for energy, and the corresponding anthropogenic impacts on climate change provide significant motivation to provide appliance-level energy consumption data to domestic consumers.

As shown in Figure 1.3a, domestic electricity use in the UK accounts for 30 per cent of aggregate electricity demand. Thus, targeting domestic consumers of electricity in an effort to reduce their consumption has the potential to reduce significantly the total carbon emissions attributed to power generation in the UK.

The IPCC find in '2013: Carbon and Other Biogeochemical Cycles' [10], that the atmospheric increase of Peta-grams Carbon (PgC) per year due to anthropogenic sources has been increasing since the late 1990's. This is illustrated in Figure 1.4. The primary contributor to this atmospheric increase is shown to be fossil fuel combustion and cement production.

The relative attribution between fossil fuel combustion and cement production is illustrated in Figure 1.5. It can be seen that emissions from the combustion of coal, oil and gas dominate the

| | 1750–2011 Cumulative PgC | 1980–1989 PgC yr$^{-1}$ | 1990–1999 PgC yr$^{-1}$ | 2000–2009 PgC yr$^{-1}$ | 2002–2011 PgC yr$^{-1}$ |
|---|---|---|---|---|---|
| Atmospheric increase[a] | 240 ± 10[f] | 3.4 ± 0.2 | 3.1 ± 0.2 | 4.0 ± 0.2 | 4.3 ± 0.2 |
| Fossil fuel combustion and cement production[b] | 375 ± 30[f] | 5.5 ± 0.4 | 6.4 ± 0.5 | 7.8 ± 0.6 | 8.3 ± 0.7 |
| Ocean-to-atmosphere flux[c] | −155 ± 30[f] | −2.0 ± 0.7 | −2.2 ± 0.7 | −2.3 ± 0.7 | −2.4 ± 0.7 |
| Land-to-atmosphere flux | 30 ± 45[f] | −0.1 ± 0.8 | −1.1 ± 0.9 | −1.5 ± 0.9 | −1.6 ± 1.0 |
| *Partitioned as follows* | | | | | |
| Net land use change[d] | 180 ± 80[f,g] | 1.4 ± 0.8 | 1.5 ± 0.8 | 1.1 ± 0.8 | 0.9 ± 0.8 |
| Residual land sink[e] | −160 ± 90[f] | −1.5 ± 1.1 | −2.6 ± 1.2 | −2.6 ± 1.2 | −2.5 ± 1.3 |

**Figure 1.4:** 'Global anthropogenic CO2 budget, accumulated since the Industrial Revolution (onset in 1750) and averaged over the 1980s, 1990s, 2000s, as well as the last 10 years until 2011. By convention, a negative ocean or land to atmosphere CO2 flux is equivalent to a gain of carbon by these reservoirs. The table does not include natural exchanges (e.g., rivers, weathering) between reservoirs. The uncertainty range of 90% confidence interval presented here differs from how uncertainties were reported in AR4 (68%).' (IPCC, 2013:p.486)

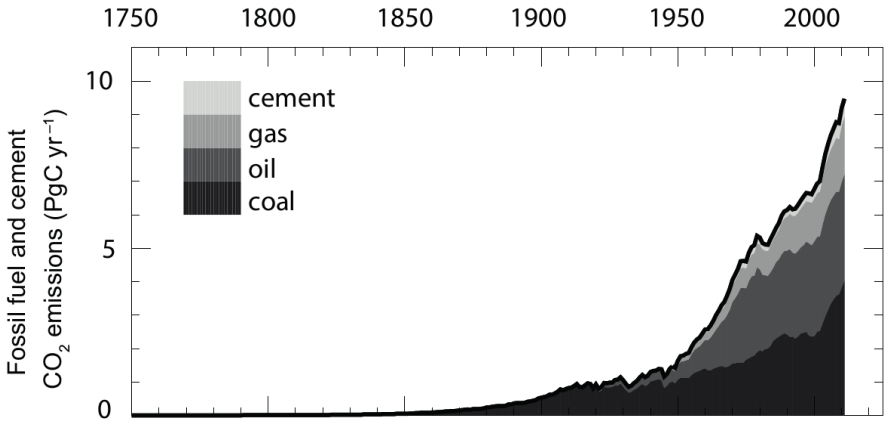



**Figure 1.5:** 'Fossil fuel and cement $CO_2$ emissions by category, estimated by the Carbon Dioxide Information Analysis Center (CDIAC) based on UN energy statistics for fossil fuel combustion and US Geological Survey for cement production (Boden et al., 2011).' (IPCC, 2013:p.487)

overall emissions, with coal foremost amongst them and demonstrating a large increase in emissions since the late 1900's.

The atmospheric increase of PgC is offset by the Earth's natural carbon sinks. About half of the emissions since the onset of industrialisation (1750 onwards) have remained in the atmosphere, the rest have been absorbed by the ocean or by other natural processes (including rock weathering and photosynthesis) [10]. Figure 1.6 shows the partitioning of the total emissions into the land and ocean sinks along with the atmospheric $CO_2$ growth rate. It demonstrates the significance of emissions attributed to fossil fuel and accelerating annual $CO_2$ emissions that are remaining in the atmosphere.

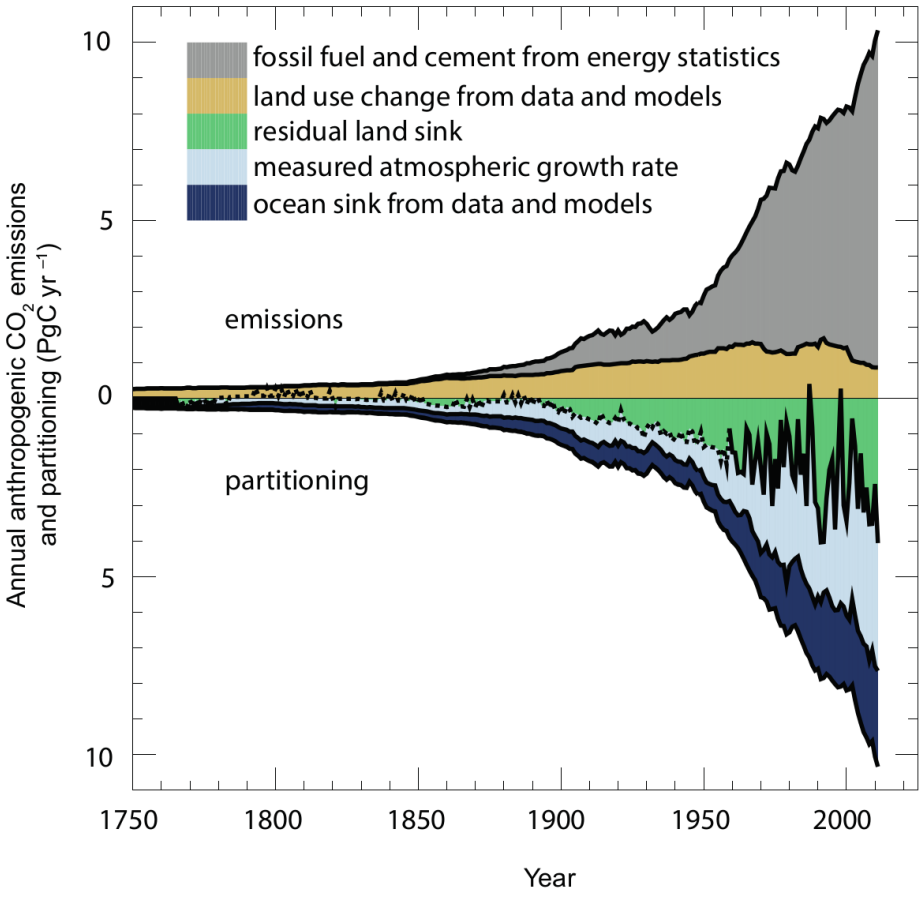



**Figure 1.6:** 'Annual anthropogenic $CO_2$ emissions and their partitioning among the atmosphere, land and ocean (PgC $yr^{-1}$ ) from 1750 to 2011.' (IPCC, 2013:p.487)

### 1.1.3 Climate Change

Substances and processes that alter the Earth's energy budget drive climate change. The change in energy fluxes caused by such substances and processes is quantified as 'Radiative Forcing' (RF). Positive RF leads to surface warming, negative RF leads to surface cooling. The units of RF are watts per square metre ($Wm^2$) [32].

It has been demonstrated that anthropogenic RF has increased more rapidly since 1970 than in previous decades. The RF from emissions of greenhouse gases (including $CO_2$) for 2011 relative to 1750 is 3.00±0.78 $Wm^2$, and emissions of $CO_2$ alone contribute 1.68±0.35 $Wm^2$. The total natural RF from changes in radiance of the sun and volcanic eruptions make only a tiny contribution to total net RF, quantified as 0.05±0.5 $Wm^2$ [10].

> 'Human influence on the climate system is clear. This is evident from the increasing greenhouse gas concentrations in the atmosphere, positive radiative forcing, observed warming, and understanding of the climate system'. [32, p. 15]

It is extremely likely that human influence is the dominant cause of observed climate warming since the mid-20th century. Human influence has been identified in warming of the atmosphere and ocean as well as in changes in the water cycle, and global reductions in snow and ice. There is also evidence for anthropogenic influence in changes in some climate extremes. Figure 1.7 illustrates the discrepancy between models excluding anthropogenic influence and including anthropogenic influence in explaining the change in large-scale climate indicators.

It is likely that by the end of this century the dry areas of the globe will experience more frequent drought, that monsoon areas will experience a lengthening monsoon season, and that precipitation will intensify in all wet regions.

In [45], Nicholls and Kebede consider the impacts of sea-level rise of between 0.5 and 2.0 m by 2100. They find the cost to protect infrastructure and population from sea-level rise attributed to climate change to be between $28bn and $90bn per year up to 2050. The authors identified significant potential threats to the UK including the disruption of supply chains as a result of more frequent coastal disasters as well as security threats due to forced population movements.

The IPCC has found that continued emissions of greenhouse gases will cause further warming and changes in the global climate system. A key factor in limiting further climate change is a global and sustained reduction in greenhouse gas emissions. [32].

The UK has passed the Climate Change Act, which commits the UK to an 80% reduction in emissions compared to 1990 levels [7]. Accomplishing this target will require a variety of approaches, including a reduction in reliance on energy generation from fossil fuels and a move toward renewable energy sources: as an island nation, key amongst these will be power generated from wind turbines.

In addition to approaches that target the means by which electricity is generated, gains can be made in educating consumers about their energy use.

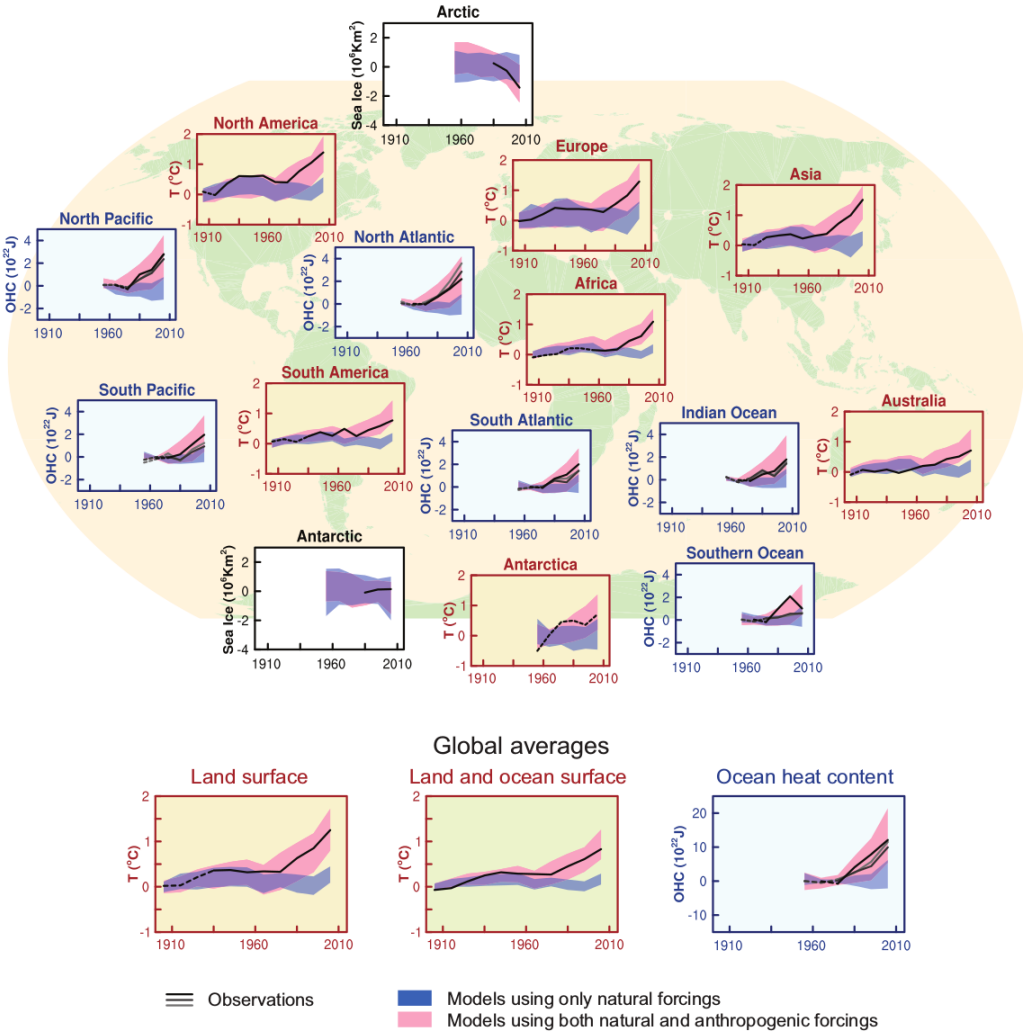



**Figure 1.7:** 'Comparison of observed and simulated climate change based on three large-scale indicators in the atmosphere, the cryosphere and the ocean: change in continental land surface air temperatures (yellow panels), Arctic and Antarctic September sea ice extent (white panels), and upper ocean heat content in the major ocean basins (blue panels).' (IPCC: Summary for Policymakers, 2013:p.18)

### 1.1.4 UK Smart Meter Programme

It is to be anticipated that every home in the UK will have a smart energy meter installed in the best case by 2020 [23], and by the latest 2032 (based on the National Grid Future Energy Scenarios

assessment [44]). However, these meters will only supply information regarding the aggregate energy usage by the home or business. Recently, Armel et. al [8] have observed that there are three broad categories of benefits from appliance-specific data over whole-home data:

> '(1) benefits to the consumer through direct feedback as well as automated personalized recommendations and more, (2) research and development benefits, and (3) utility and policy benefits.'

Focussing primarily on residential applications, the authors find that annual percentage energy savings of between 3.8% and 8.4% should be achievable using 'indirect' feedback (that is feedback provided after energy consumption occurs). These figures rise to between 9.2% and 12.0% when used in combination with real-time appliance level information.

Armel et. al [8] note that the impact of smart appliances (that is to say, appliances capable of providing the user with their own energy consumption data) is uncertain. They assert that such smart appliances would likely be limited to white goods, and that the typical turn-over time for this market is 12 years. These two factors limit the energy savings opportunities available for the consumer as a result of smart appliances.

In the UK Government's response to the consultation on the second version of the Smart Metering Equipment Technical Specifications, consideration is given to the likely proliferation of 'Consumer Access Devices' with the advent of smart meters. A Consumer Access Device is any device that can be connected to a smart metering system in order to provide a consumer with information about their energy use.

The UK Government believes that access to such information would empower consumers and allow them to better manage their energy use, as well as allow for wider adoption of demand response technologies [21].

### 1.1.5 Summary

The UK faces uncertainty in its provision of electricity without loss of load in the short to medium term, as a result of demand for electricity and possible shortage of supply. We have seen that domestic electricity makes up 30% of aggregate demand within the UK, and that a majority of electricity within the UK is generated by fossil fuels, which contribute to the Peta-grams of Carbon released globally each year . There is clear evidence that this Carbon has an influence on the climate system. The UK will be introducing mandatory smart-meters into homes in the very near future. These meters provide a new opportunity to empower consumers with richer information on their electricity usage. It has been shown that consumers are better able to reduce their energy consumption when armed with appliance level information. Non-Intrusive load monitoring may provide the means to break down household electrical demand into appliance level data, and help people to change their habits and recycle their old, inefficient appliances.

## 1.2 History of Non-Intrusive Load Monitoring

A large and growing body of literature has investigated the practicalities of disaggregating appliance level energy consumption from aggregate whole-home data. This is commonly referred to as Non-Intrusive Load Monitoring (NILM).

In his 1989 paper on the subject [26], George W. Hart observed that merely using laboratory measurements of each appliance's energy consumption is not useful, since the majority of appliance characteristics are usage dependent. Consider a 2kW hair-dryer: each home may be expected to contain at least one such device. However, the amount of time spent using this device will depend very strongly on the type of occupants in the home. A household containing a majority of long haired occupants would find significantly more utility in the hair-dryer than a household containing mainly bald occupants!

Whilst it is possible to perform exhaustive monitoring of devices within a residence or business, this is very intrusive to the home or business owner and has significant issues. Hart observed that

the intrusive approach has issues 'beyond the obvious time, cost, inconvenience, and damage associated with appliance surveys, sensor installation, data collection, sensor maintenance, and eventual sensor removal' [26]. Clearly the devices within a home can and do change as a function of time. This renders surveys inaccurate beyond a small window after they are conducted. Additionally, a user that willingly submits to a full appliance survey or intrusive installation of load monitors on every power outlet in their home may well be more energy conscious than the average. Thus, the information provided may be of less benefit than if it were provided to a more profligate consumer for whom greater savings or efficiency may be possible. Neither does the intrusive approach scale to a national or international level - who would be willing to pick up the cost of installation of many tens of millions of individual appliance load monitors? How would the public react to proposals of legislation enforcing access to their homes to install such monitors?

Hart proposed a non-intrusive technique designed to eliminate such problems by modelling the household power wiring as a communication channel and treating the power flows into the home as signals to be analysed for their information content. This requires only a single monitoring device on the main supply of power to the home [26].

Hart's original approach made use of the step changes in the total power consumption of the house to detect on/off state changes in an appliance. Hart makes use of both Real and Reactive Power.

In an alternating current (AC) supply, Real Power is drawn by purely resistive loads. Loads that draw Real Power result in both current and voltage polarities reversing at the same time. Some devices also draw Reactive Power. In the Reactive component of power, voltage and current are up to 90 degrees out of phase with each other.

Devices can be characterised by considering both their Real and Reactive power in the Complex plane. Complex Power is the vector sum of Real and Reactive Power.

In Hart's 1989 work, once step changes in both Real and Reactive have been recorded from the whole-home monitor, they are clustered into regions. The visualisation of these regions is shown in Figure 1.8.

In his 1992 work, Hart discusses refinements to his original approach and describes a method of disaggregation based upon a total load model which assumes a linear appliance model and associates a time-invariant complex power with each appliance. Purely resistive appliances are associated with an imaginary power of zero (since there is no Reactive component of power in such appliances) [27].

Hart identifies three general classes of appliance models:

- **ON/OFF**
  Appliances that are limited to two states: ON and OFF. There exists only a single ON state.
- **Finite State Machine (FSM)**
  Appliances that may be modelled with an arbitrary set of discrete states and transitions between these states. One example of such an appliance might be a toaster with multiple intensity settings.
- **Continuously Variable**
  Appliances (such as dimmer switches) that have an effectively infinite number of states.

Hart's 1992 work constructed a prototype Non-Intrusive Load Monitor (NILM) that modelled only ON/OFF devices. He observed that this gave satisfactory results, and in his tests was able to disaggregate 86% of the total energy consumed in a residence into the responsible appliances. Hart noted in his paper that the modelling of devices as FSMs and using FSM algorithms should be a priority in future research since there exist many appliances that are best modelled in this manner.

Hart provides an important insight into the typical power consumption in U.S homes. His breakdown of household power consumption is shown in Figure 1.9. One can see from this figure that the majority of energy consumed in a household (in 1992) was used for appliances responsible for either heating or cooling the home, or for heating water. It can be seen that such appliances typically consume power at a rate of >1000W. Hart [27] notes that his prototype NILM was capable

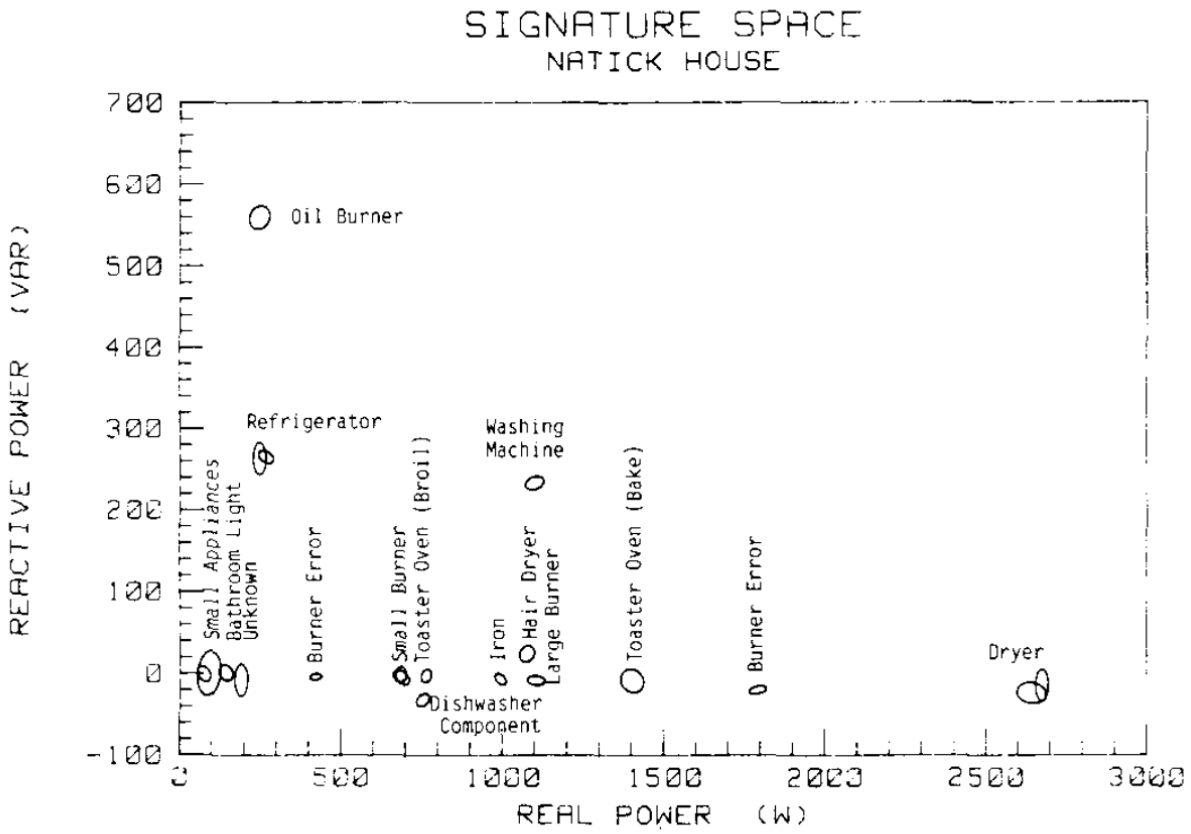



**Figure 1.8:** 'Regions of the complex power plane associated with appliances' (Hart, George W,1989:p.13)

of disaggregating devices that consume >150W reasonably reliably but failed below this threshold.

As observed by Norford and Leeb in their 1996 work [46] the proliferation of multi-state devices, drawing variable power, since the introduction of NILM, as well as the decrease in power drawn by individual devices has increased the difficulty in accurately modelling and metering all appliances. This has spurred research into methods by which such devices may be modelled and disaggregated.

However, Bergés et al. (2010) still find that almost 80% of the total household consumption in the U.S. is down to the top 12 energy consuming devices in the home [6].

### 1.2.1 Evolution

Zeifman and Roth (2011) [56] found that the landscape of NILM devices is now split into two types of devices: Low-frequency hardware and Higher-frequency sampling hardware.

High-frequency hardware is considered to sample at rates at or above 1Hz. Since the fundamental period in A/C supplies is either 50 or 60Hz, devices using sampling frequencies in the second or multi-second range are considered Low-frequency.

Zeifman and Roth (2011) identified that Hart's approach can relatively easily detect ON/OFF type appliances, but has apparent problems detecting multi-state and variable load appliances. Importantly, Sultanem (1991) finds that many appliances change their resistance as they turn on, meaning that there can be a mismatch in turn-on power to turn-off power of as high as 10% [54] which poses issues for Hart's clustering method if it is not modified to take this into account.

Cole and Albicki (1998) [11] have extended Hart's method (henceforth referred to as the 'MIT method', due to its origins at that institution) by considering the the characterisation of the edges and slopes of each appliance's power signature, in addition to its steady-state power draw.

Cole and Albicki state in [11]:

| Appliance | Power (W) | Energy (TWH) |
|---|---|---|
| Refrigerator | 200* | 145.1 |
| Resistance Heat | ≥ 2000* | 137.3 |
| Water Heater | ≥ 2000* | 128. |
| Lighting | 75* | 100.* |
| Central AC | ≥ 2000* | 82. |
| Heat Pump | ≥ 2000* | 45.9 |
| Freezers | 250* | 44.1 |
| Stove/Oven | 1800 | 40. |
| Furnace Fan | 500 | 22.5 |
| Color TV | 193 | 21.8 |
| Window AC | 860 | 18.8 |
| Dishwasher | 1200 | 17.9 |
| Central Heat Pump AC | ≥ 2000* | 17.5 |
| Waterbed | 347 | 12.6 |
| Microwave | 1450 | 8.6 |
| Pool pump | 1008 | 6.0 |
| Aquarium | 100* | 5.5 |
| Crankcase heater | 100* | 5.4 |
| Hot tub | 675 | 4.6 |
| Clock | 2 | 4.5 |
| Well pump | 1300 | 4.4 |
| Dehumidifier | 257 | 4.4 |
| Toaster oven | 1146 | 4.3 |
| Audio | 71 | 4.1 |
| Hair Dryer | 600 | 3.4 |
| Electric blanket | 177 | 3.2 |
| Vacuum | 630 | 2.7 |
| Grow light | 40 | 2.4 |
| VCR | 20 | 2.4 |
| Coffee maker | 1200 | 1.8 |
| B+W TV | 73 | 1.8 |
| Computer | 100* | 1.7 |
| Iron | 950 | 1.6 |
| Humidifier | 177 | 1.1 |
| Engine heater | 100* | 1.0 |
| Ceiling fan | 88 | .9 |
| Exhaust fan | 200 | .8 |
| Attic fan | 370 | .6 |
| Other | 100* | 2.0 |
| Total | | 914.5 |

**Figure 1.9:** 'Typical Power Consumption and Estimated Annual U.S Residential Energy Consumption of Household Appliances' (Hart, George W, 1992: p.1888)

> 'All of the complex residential loads investigated have turn-on's that can be characterized by three phases: an initial upward spike in power, slower changing variations, and a settled power level. These three phases will be called edges, slopes, and steady-states respectively.'

The features they discuss can be seen in the two device profiles that are shown in Figure 1.11.

Baranski and Voss (2004) explored the use of a genetic algorithm for device detection in NILM systems [2] based upon a 1Hz sampling rate. This approach has the advantage that it can detect frequently occurring patterns in a load trace without any 'a priori' knowledge concerning the nature of the appliances. The approaches thus far described have required a signature 'library' of devices to be established in order that disaggregation can be performed. Another novel feature of their approach is that it does not require the measurement of both active and reactive power. This reduces the cost of implementation of the system as the reactive power measurement sensors are more costly than those for active power alone [56]. Baranski and Voss use a fuzzy clustering method and treat detectable appliances as finite state machines with a fixed number of states and assume that those states have a fixed order (i.e. The graph describing the FSM is directed) [2].

Zeifman and Roth (2011) find that most researchers agree that in order to reach a high accuracy

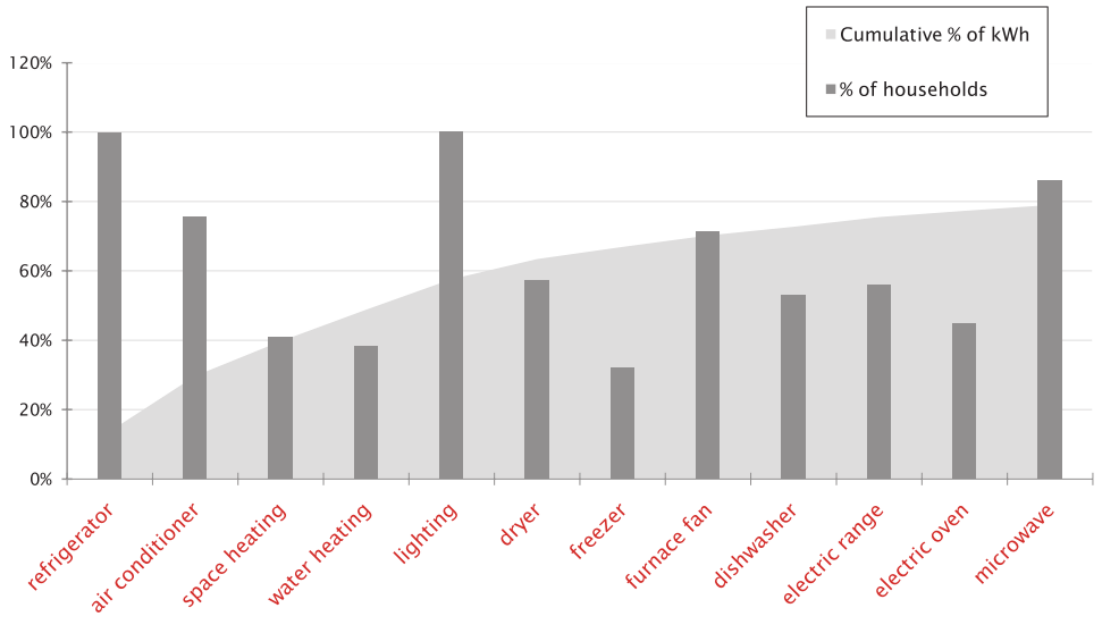



**Figure 1.10:** 'Top 12 electricity-consuming appliances in the United States, ordered by their average annual kilowatt hour (kWh) use (EIA 2001).' (Bergés et al., 2010: p.847)

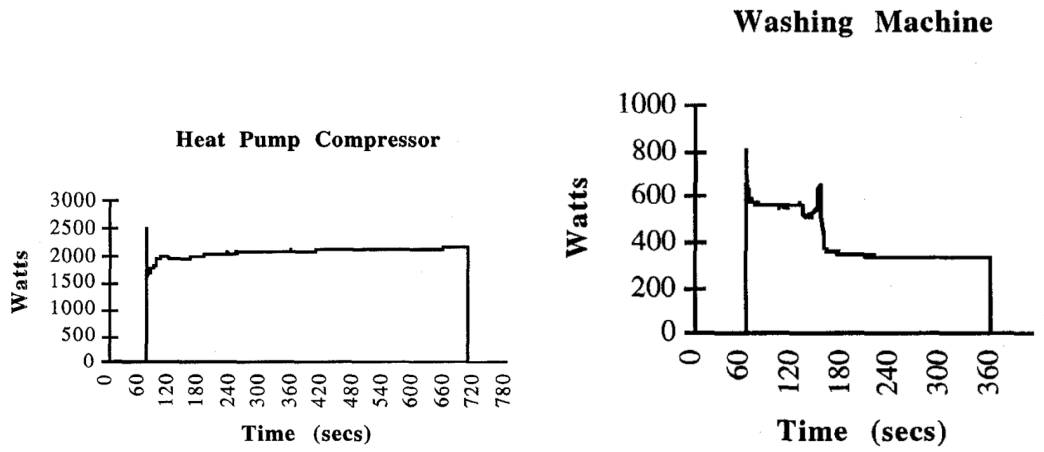



**Figure 1.11:** 'Real power consumption of one operation cycle for a heat pump compressor and a washing machine' (Cole and Albicki, 1998a: p.813)

of detection of appliances in a NILM system, higher sampling rates are required [56]. This is because a higher sampling rate allows for the detection of further harmonics and signal waveforms that may be produced by devices. Lower sampling rates only allow for the detection of relatively coarse ON/OFF state changes or macroscopic transients (i.e. transients that last on the order of seconds).

The authors do observe, however, that the use of higher sampling rates causes issues with data transmission and storage. This may necessitate the caching of data on-board the monitoring device, or place a significant burden on the communication network if the monitoring device is attempting to stream the data to a remote machine. Smart Meters have limited computational capacity and most commonly use the ZigBee standard for wireless transmission which has a typical bandwidth of 250kbps [8]. When considering higher sampling rates, these limitations must be taken into account.

Harmonics and spectral envelopes have been investigated as an addition to steady-state event detection. Wichakool et. al (2009) find that the inclusion of device harmonic data allows their algorithm to work with a larger set of loads than would otherwise be possible. In their 2009 paper, the authors find that they are also able to track multiple variable-state devices in a non-

intrusive manner using harmonic and spectral data [55]. However, Zeifman and Roth (2011) find that excessive training is required for each particular device before classification and monitoring can be performed. They also observe that the performance accuracy remains uncharacterised for many practical scenarios, and that it is not known how the presence of a new device without a known signature affects the algorithm's performance [56].

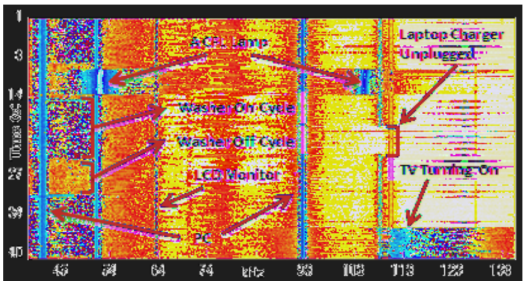



**Figure 1.12:** 'Frequency spectrogram showing device actuation in a home' (Gupta, Reynolds and Patel, 2010: p.141)

Electromagnetic interference (EMI) has also been considered by Gupta, Reynolds and Patel (2010) as a potential method for distinguishing devices in a NILM system [24]. However, Zeifman and Roth [56] observe that the authors of [24] admit that the EMI signatures emitted by devices may depend on the wiring layout of the home, and thus such a method of detection may not be scalable. Additionally, the authors of [56] observe that overlaps in signatures will certainly occur, and furthermore, that purely resistive devices such as stoves and dryers do not even emit detectable EMI. This places limitations on the use of such detection technology. It must also be considered that the practicality of such a system is limited since the use of EMI signatures for disaggregation does not allow for any estimate to be made of the energy consumption of the devices [56].

### 1.2.2 Algorithms

The algorithmic approaches to disaggregation in NILM systems are broadly divided into two major approaches [56]:

- **Combinatorial Optimisation**
  In this approach a combination of known appliance models is sought such that the resultant modelled aggregate signal is as close to the observed aggregate signal as possible. One such means to do this is using a least residue (LR) algorithm. A LR algorithm is typically used when the unknown signal to be identified represents only one appliance being switched on or off at one time and not a composite load. In this case the residue between the unknown appliance and known appliance signatures from a database is computed. The entry that corresponds to the minimum residue is selected as the solution [40].

  As discussed by Liang et al. (2010) the optimization problem becomes more complex when the composite load to be disaggregated represents a snapshot containing multiple appliance signatures. Integer programming techniques or genetic algorithms can still be applied to solve such problems at the cost of greater computational burden [40].

- **Pattern Recognition Approach**
  This approach involves matching detected state changes to a known library of features representing the states of each appliance. Techniques that have been used to do this include nearest-neighbour, Bayesian classification and neural networks [56]. Bergés et al. used a variety of different off-the-shelf machine-learning algorithms to compare classification accuracy of newly detected events against a library of known appliance state transitions [6]. The authors of [6] found that a 1-nearest neighbour algorithm provided acceptable classification accuracy. However, in their paper the sampling rate used was 10kHz and thus this method may not apply to data at the super-second sampling rates likely to be available from UK Smart Meters [23].

Pattern recognition techniques have the advantage that they can be used to identify loads in a whole-home signal even if not all appliance signatures are known. Combinatorial optimization techniques suffer the issue that the presence of unknown loads complicates the optimization problem as the solutions that are found are based only upon combinations of known appliances [57].

Zoha et al. (2012) observed that techniques such as Artificial Neural Networks (ANNs) and the use of Hidden Markov Models (HMMs) have been shown to perform well, due to their ability to incorporate both temporal as well as appliance state transition information. [57]. The authors of [57] do note, however, that since the complexity of HMMs exponentially increase as each device is added, there are limits to the applicability of this method in attempting complete disaggregation beyond the top 12 energy consumers in a home. However, since Bergés et al. (2010) found that 80% of the energy consumption in a U.S. home is typically explained by the top 12 devices [6], HMMs or variants thereof may still provide a valuable contribution to the disaggregation problem. HMMs have the additional benefit that they allow other input information, such as temporal and temperature data, to be included in the models.

Zoha et al. (2012) propose that NILM research is moving towards a multi-modal sensing framework that could allow for better appliance models and address major challenges in disaggregation. This would allow the incorporation of environmental sensing data (including temperature and occupancy monitoring).

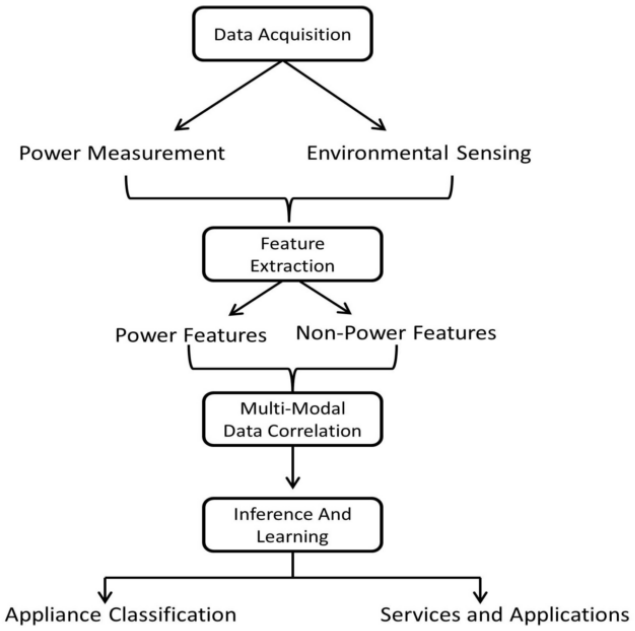



**Figure 1.13:** 'Multi-Modal Sensing Framwork for NILM based load disaggregation' (Zoha et al., 2012: p.16859)

Researchers have recently started to use unsupervised learning techniques in order to achieve disaggregation in NILM systems without the need for a-priori information (such as the appliances present in a home). These techniques are desirable since the algorithms do not have a training requirement, and have minimal set up cost [57].

Zoha et. al (2012) provide details of the current varieties of learning algorithms but they observe that:

> 'Most of the research work in NILM report the performance of their system using accuracy metrics. However, due to inconsistency in the definition of accuracy it is not possible to draw meaningful comparisons between reported research work.'

Batra et al. (2014) have also observed that the lack of available reference implementations of state-of-the-art disaggregation algorithms leads to authors often comparing their work against more basic benchmarks [4]. In the authors' words:

| Learning Algorithm | Features St [a]/Tr [b] | Accuracy % | Training S [c]/U [d] | Online/ Offline | Scalability | Appliance Types |
|---|---|---|---|---|---|---|
| SVM [11,17,33,54] | B [e] | 75–98 | S | Online | Yes | I, II, III & IV |
| Bayes [12,50,54] | St | 80–99 | S | B | No | I & II |
| HMM [49,59,60] | St | 75–95 | B | Offline | No | I & II |
| Neural Networks [17,37,61] | B | 80–97 | S | Online | Yes | I & II & III |
| KNN [6,9,62] | B | 70–90 | S | B | Yes | I & II |
| Optimization [7,18,20,35] | St | 60–97 | S | Offline | No | I & II |

[a] Steady-State [b] Transient [c] Supervised [d] Unsupervised [e] Both.

**Figure 1.14:** 'Comparison of load disaggregation algorithms' (Zoha et al., 2012: p.16855)

> 'This problem is further compounded since there is no single consensus on which benchmarks to use, and as a result most publications use a different benchmark algorithm.'

Batra et al. have taken steps to address this with the introduction of NILMTK, which provides a number of accuracy metrics to enable comparison of different algorithm's accuracy at disaggregation. NILMTK currently contains implementations of two benchmark disaggregation algorithms [4].

Kolter and Jaakkola (2012) incorporated behavioural patterns such as time of day and duration of use into their improved additive factorial HMM to constrain the optimization problem and achieved results beyond that which had previously been possible with hourly data [38], [8].

In 2009 and 2010 Bergés et al. used a competition strategy among multiple algorithms to determine the best match for each appliance [5] [6]. Armel et al. (2013) propose that 1s-1min data may be sufficient for disaggregation, particularly if appliance recognition can be augmented with improved algorithms [8].

## 1.3 Project Aims

Given that the UK Smart Meter draft specification implies sampling rates in >1s range [23], it seems that further investigation of benchmark algorithms to determine the most effective that operate in this sampling range is an appropriate direction for further research.

The author plans to attempt an implementation of Hart's edge-detection algorithm as described in [27], within the framework of NILMTK (this implementation might also be extended to include a probabilistic analysis based upon ambient temperature and occupancy monitoring). This seems an appropriate avenue for further work, given the observations made by Zoha et al. and Batra et al. ( [57] and [4]) that a lack of benchmark algorithms and accuracy metrics makes comparison of NILM approaches very difficult at present.

## 1.4 Dissertation Outline

**Chapter 2** describes the hardware and software used to monitor temperature, occupancy and power data.

**Chapter 3** describes the first stages implemented in the disaggregation system.

**Section 3.2** describes the preparation and normalisation of data.

**Section 3.3** describes the edge-detection algorithm.

**Section 3.4** describes the process of pairing state-transitions that are detected.

**Chapter 4** evaluates clustering methods and describes how power-states are clustered in the signature space.

**Chapter 5** describes two approaches to matching clustered data to appliances. The design has some success in disaggregating several different types of devices.

**Chapter 6** discusses the relationships found between temperature, occupancy, and appliance use.

**Chapter 7** discusses conclusions, limitations of the implementation and further work.

## 1.5 Terms Used In This Report

- **Active (Real) Power**

  Active Power, $P(t)$, also known as Real or True Power, is power in an AC circuit that actually performs work. It is measured in Watts. Purely resistive devices do not introduce a phase shift between the voltage and current in a circuit, and result in only Active Power being drawn.

- **Reactive Power**

  Reactive Power, $Q(t)$, can be thought of as counter-acting the effects of true power. Purely capacitative loads cause the voltage waveform to lag the current waveform in an AC supply by up to 90°and thus result in negative reactive power. Purely inductive loads cause the voltage to lead the current waveform by up to 90°and thus result in positive reactive power. Reactive power is measured in Volt-Amperes-Reactive - VAR.

- **Apparent Power**

  Apparent Power, $S(t)$, is the magnitude of the vector sum of the Active and Reactive Power, and is determined by the AC Power Triangle, shown in Figure 1.15. It has units of Volt-Amperes - VA.

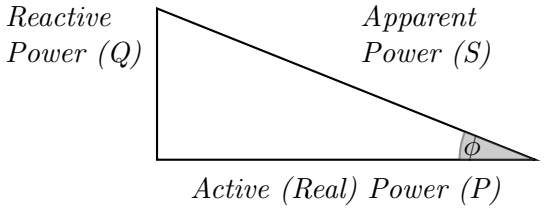



**Figure 1.15:** The AC Power Triangle

  All practical electrical devices have some combination of resistive, capacitative and inductive elements, and thus the power measured on the whole-home circuit will have some combination of both Active and Reactive Power, depending on the appliances in use. Appliances themselves will each have some combination of Active and Reactive Power.

- **Normalised Power**

  We define Normalised Power, $P_{Norm}(t)$ , where $P(t)$ and $V(t)$ are Power and Voltage respectively - as follows:

$$P_{Norm}(t) = \left(\frac{240}{V(t)}\right)^2 P(t) \tag{1.1}$$

  Normalised Power can be thought of as what the power *would be* if the utility provided a steady 240V supply to the home, and if loads obeyed a linear model.

- **Power-State / Steady-State** We define a Power-State as a period of time exceeding our minimum sample period (a figure of 2 seconds has been used in the implementation described) for which we have detected a matching start and end transition (*Edge*), and during which the power does not vary by more than an allowable tolerance (15W has been used in the implementation).

- **Transient**

  We define a transient as a short-lived period during which the power changes rapidly. This is determined by reference to the same tolerance as used in the Steady-State definition. If the fluctuation of power of multiple seconds is greater than 15W between each sample, then we are in a transient period.

  A good example of a device exhibiting this behaviour is the refrigerator compressor, which has a large transient spike in power draw when it first engages, but then settles into a steady-state after a short period. This is illustrated in Figure 1.16.

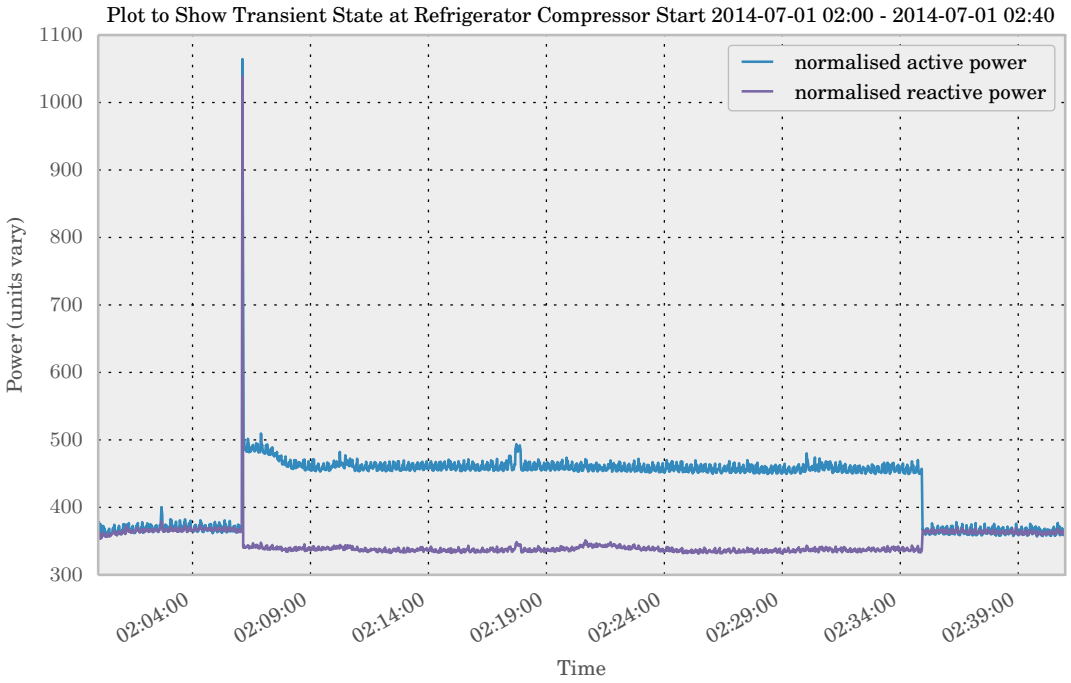



**Figure 1.16:** Visualising the transient state of the refrigerator compressor start

- **Edge**

  We define an Edge as the transition between two steady power states in the whole-home signal. An Edge represents the difference in the average power of two steady states. The Edge detection algorithm that has been implemented 'rides over' transients, that is to say, periods where the power is changing rapidly from second-to-second. An Edge will be output from the algorithm when a steady state is next reached after a transient period. An Edge consists of a 2-tuple, containing the change in active and reactive power between the preceding and current state. For example:

$$SteadyState_1 = (300, 100)$$
$$SteadyState_2 = (500, 50)$$

$$\Delta_{1\to2} = SteadyState_2 - SteadyState_1$$
$$\Delta_{1\to2} = (200, -50)$$

- **Signature Space**

  We define our Signature Space as the n-dimensional space represented by the characteristic quanta of our measurements of power. In this case, we use normalised active and reactive power measurements to quantify the power-state of an appliance, thus our signature space is 2-dimensional. Appliance power-states are detected as multi-variate Gaussian distributions in the signature space. Figure 1.17 shows an example of an individual cluster representing a power-state within the signature space, distributed as a multi-variate Gaussian, with an ellipse drawn at the 2-standard deviation interval from the cluster centroid.

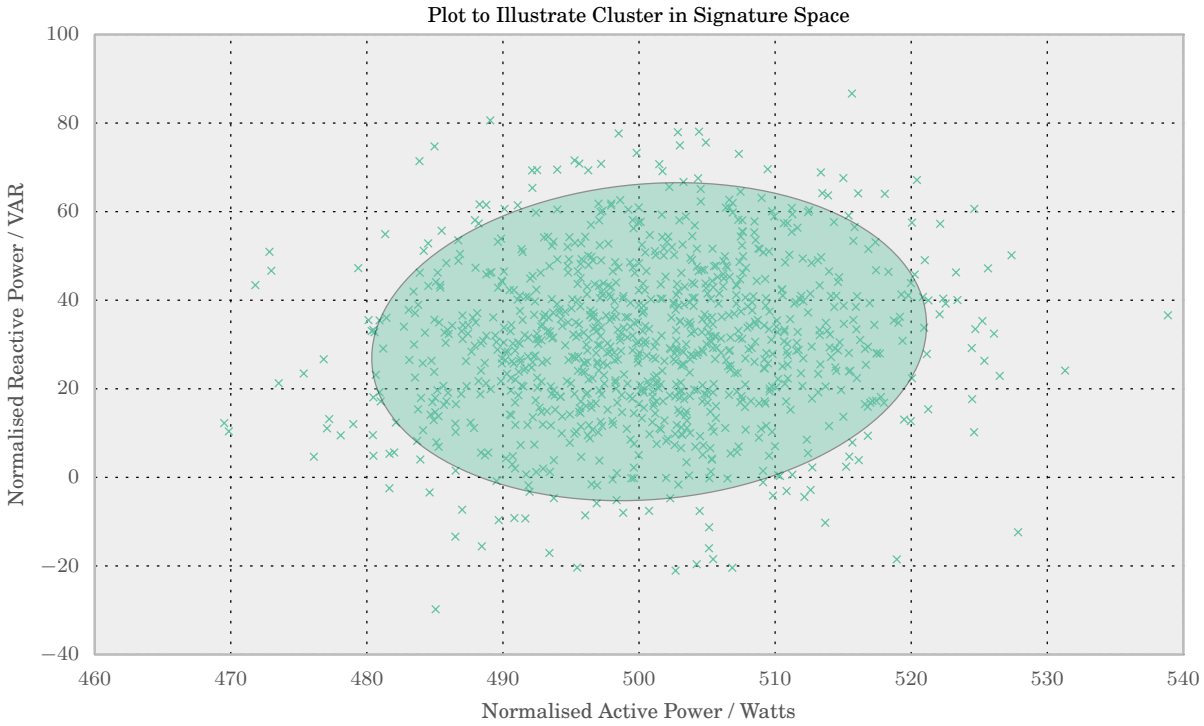



**Figure 1.17:** An idealised multi-variate Gaussian cluster in the Active/Reactive signature space, generated by sampling from a distribution parametrised with centroid: $(500, 30)$ and covariance matrix: $\begin{bmatrix} 100 & 20 \\ 20 & 350 \end{bmatrix}$.

## 1.6 Technologies Used

The algorithms that have been implemented use modern, open-source technologies in order to provide a benchmark disaggregation framework based heavily upon the non-intrusive load monitor developed by George Hart. The approach used by Hart has been extended and designed from the outset to be integrated into the `NILMTK` toolkit developed by Batra, Kelly and Parson [4]. The software is able to import data stored in the format used in the UK-DALE [36] dataset, and has been tested on data recorded in the author's home as well as on homes 1 and 2 from the UK-DALE dataset.

Python was selected as the language in which to implement the project. This selection was based on the desire to easily implement the work into `NILMTK`, and the availability of excellent open-source scientific packages that enable powerful analysis of time-series data. It was also a desire of the author to produce a significant software project and gain more experience using Python. Packages that have been used in the implementation include `statsmodels` [52], `scikit-learn` [49], `scipy/numpy` [34] and `pandas` [42]. These packages form a powerful suite in which to perform significant data manipulation and computation.

EMACS was used as the primary text editor and syntax checker for the project.

`IPython` [50], an interactive computing environment, was used to test ideas, keep an annotated

record of work undertaken, and to enable fast visualization of data. The use of `IPython` enables rapid prototyping of ideas and easy integration with popular data visualisation toolkits, such as `matplotlib`. `IPython` provides a browser-based notebook, in which code can be written, executed and debugged, as well as allowing for embedded visualisations such as graphs and tables. `IPython` was also used for debugging and testing functions and classes. It provides an in-line debugger that can be used to step backward and forward in the stack and query variables and function arguments, much like GDB. When dealing with algorithms and functions that have thousands of lines of input and output data, being able to immediately produce graphical output was an incredible aid in debugging, as the raw data output is very difficult for a human to parse intelligibly! Figure 1.18 shows an example of the browser-based interface.

`matplotlib` [29] is a Python plotting library that enables the generation of a wide variety of charts and graph types, and it provides an object oriented interface and a variety of classes that may be manipulated and used to create custom visualisations. It is integrated with `pandas` and was used to generate the vast majority of the visualisations shown throughout this report.

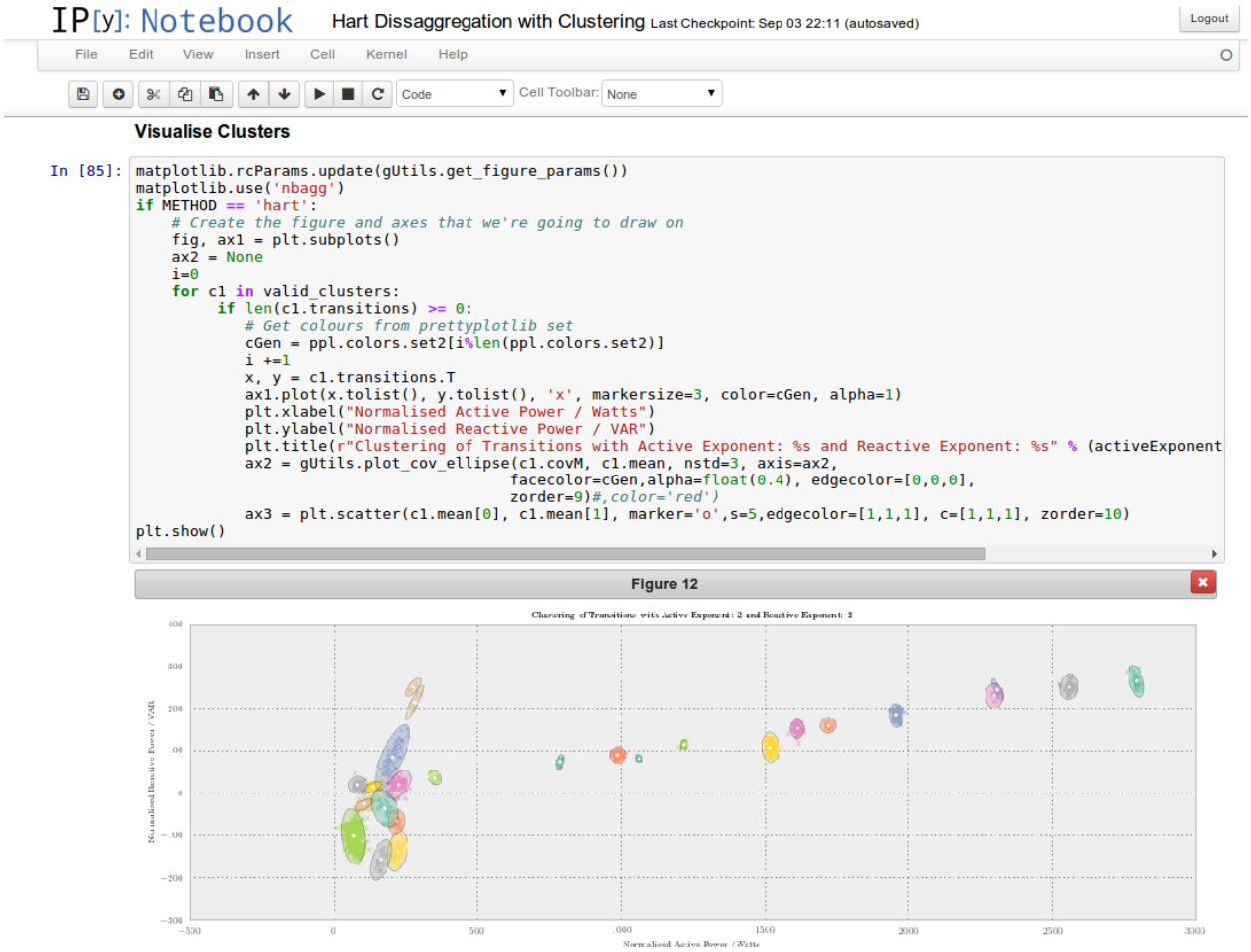



**Figure 1.18:** An IPython session showing code being executed to produce interactive graphics in a browser-based notebook.

# Chapter 2

# Monitoring the Household

## 2.1 Introduction

Whilst multiple household consumption datasets already exist (including REDD, UKDALE, IAWE) [37], [36], [3], it was decided to collect data in the author's home on the power consumption of individual devices and whole home supply in order to assess the relationship between household occupancy, temperature and device usage.

The household to be monitored is a modern, well insulated, 3 bedroom, 13th floor apartment, with two occupants.

## 2.2 Temperature Monitoring

In order to monitor the temperature in the household, sensors were investigated that allowed for the collection of temperature data at regular intervals and the extraction of this data in a machine readable format.

At the suggestion of Dr. Knottenbelt, two WiFi enabled logging devices were procured. It was suggested that WiFi and 'Cloud' enabled devices would require minimum maintenance and allow for the collection of data in the least invasive fashion.

The devices used are two Corintech WiFi-T+ high accuracy temperature sensors. The sensors are initially connected to a PC in order to link them to the home WiFi network.

Once this is complete, the sensors are able to transmit data to the manufacturer's 'Cloud' at a sample interval chosen by the user (between every 10s and every 20s). A time interval of 1 minute was selected as it was not envisaged that further time resolution would add value.

One device was then placed in a shaded location on the exterior of the dwelling, the other was placed in the author's kitchen, away from any direct sources of heat (such as the cooker or hob).

The manufacturer enables simple charting of the data on their website, as well as extraction of the 1 minute data as .CSV files, which enables easy importing of the data into other applications.


**Figure 2.1:** Externally mounted temperature sensor

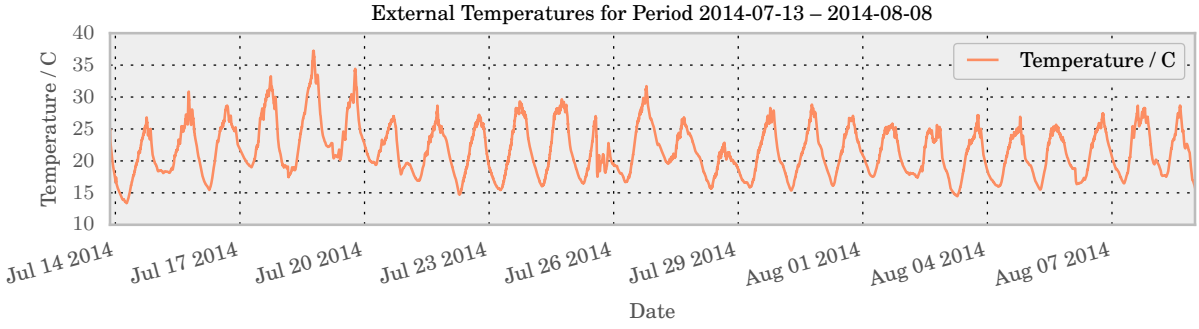
A significant issue was experienced with one of the data loggers. The logger would work correctly and transmit data for a period of time, before disconnecting from the local WiFi and failing to reconnect. Any data captured by the device after this would be lost, as a reset was required to

re-enable the device. A beta firmware was eventually obtained from the manufacturer which fixed the issue. As a result of the problems, there are gaps in the period monitored due to the lost data.

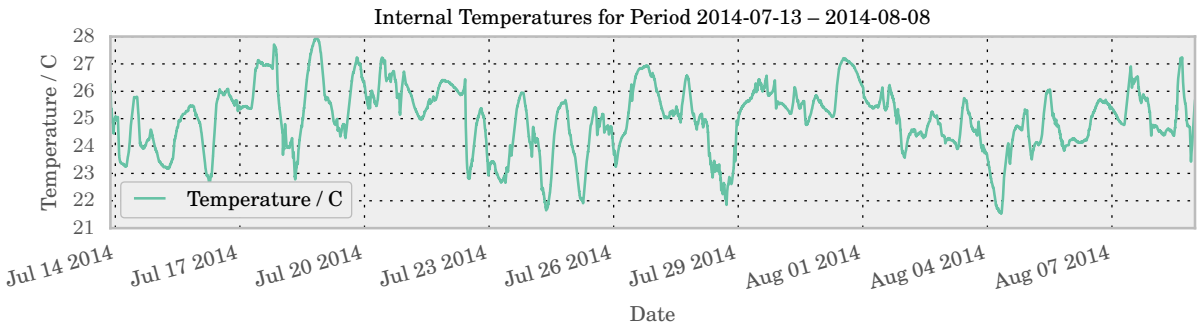
One shortcoming of the loggers is the necessity to either use the manufacturer's Cloud based logging, or to manually connect via USB to the devices and use a proprietary application to extract the data. It would be preferable for backup and potential 'live logging' scenarios to receive direct transmissions from the devices. The author attempted to packet-sniff the transmissions being sent from the device over unsecured WiFi in order to reverse engineer the protocol and directly capture the temperature data. Unfortunately the devices establish an encrypted connection with the Cloud negating this approach. Additionally, since the USB connection on the devices actually emulates an RS232 serial port, attempts were also made to intercept the communication between the devices and the manufacturers software, to investigate the possibility of streaming data from them in this fashion. Again, a proprietary communication protocol is used. Unfortunately, the manufacturer of the chips used in this product only allow registered licensees to view the data-sheet and specifications for this protocol, so it did not prove possible to stream live data directly to a PC.

The desire to stream this live data arose from planned implementation of a dynamic disaggregation algorithm (ie, using data as it arrived, rather than a purely static analysis of large data sets). In spite of the lack of live streaming temperature data, useful analysis can be performed on the data set post-collection which may inform future disaggregation efforts.

Figures 2.2 shows the data recorded by the loggers. In Figure 2.2a, the external sensor, the day/night cycle is clearly visible, along with the warm period in mid July. Dips in the temperature on an intra-day basis show the changing weather conditions. Figure 2.2b, the internal sensor, shows the general temperature trend over the period, but is significantly more complex. The temperature exhibits a hysteresis due to the insulated environment in which it is situated. Dramatic drops in temperature are also shown on an intra-day basis. Usually this is due to a balcony door or window being opened which then tends to equalise the internal and external temperature.




**(a)** External Temperatures




**(b)** Internal Temperatures

**Figure 2.2:** External and Internal temperatures recorded for the period 13/07/2014 - 08/08/2014

## 2.3 Occupancy Monitoring

In an effort to investigate the relationship between occupancy and device usage, a simple occupancy monitor was developed. A variety of direct and indirect methods were considered.

Passive Infrared (PIR) sensors are typically used in burglar-alarm systems to detect an occupant in a room. Unfortunately, they require that there be some motion in order to record an occupant. This is not appropriate for 'extended' occupancy monitoring as household members will of course sit for extended periods to watch television, or use a computer. Additionally, the installation of PIR sensors requires a physical modification to the dwelling, and thus was discounted for its invasive nature.

RFID tags for occupants were also considered, but discounted due to the requirement to keep them on one's person. It is to be expected that any 'extra' device would be forgotten on occasion, and thus render the occupancy monitoring inaccurate.

Eventually, it was decided to use an 'indirect' measure of occupancy. Given the ubiquity of mobile phones, and the fact that they are invariably co-located with their owner, it was thought that detecting the presence of a mobile phone provided a highly accuracy proxy for the presence of an occupant. It also enables the presence of individual occupants to be determined and thus gives additional information.

Bluetooth technology requires that every device have a unique hardware address. Thus, if the hardware addresses of the Bluetooth adapters within occupants' mobile phones are known, they can be used to build an occupancy monitoring system.

An extremely important property of the Bluetooth protocol is that even if a device is not in 'announce' mode (in which it advertises its presence and capabilities to other nearby devices) it can still be directly polled by its hardware address, and will respond to this poll. It is this property that enables a car-audio system to reconnect to a phone once the driver enters the vicinity of the car, without any intervention on the part of the driver.

Rob Collingridge's [12] description of how to ascertain the hardware addresses of nearby Bluetooth devices was used along with a significantly modified Bash script from Nick Stallman's similar Bluetooth project [53] in order to pair with, and then poll for the presence of both occupants' phones.

Once the phones have been 'paired', the script uses a Belkin F8T017 USB Bluetooth adapter connected to an Ubuntu 12.04 Server to continuously check for the presence of any number of phones. If the server is restarted or loses power, a cron-job ensures that the script is restarted and continues to monitor for occupancy. The script outputs to a text file the date and time, and whether the paired phones are detectable or out of range. Tests were performed which indicate that all rooms in the household are within range. However as with any wireless technology, small drop-outs do occur. These are detectable as very short periods (between seconds and minutes) where a phone is not seen as present. Data analysis allows for these to be detected and compensated for.


**Figure 2.3:** Belkin F8T017 Bluetooth adapter used to monitor occupancy

In order to ensure that Bluetooth was enabled at all times on each phone, 'Tasker' [18] software was used to enable the Bluetooth stack on each phone when the phones were within range of either local cell towers or the household's WiFi access point. Figure 2.4 shows the occupancy data recorded for a period of 8 days at the beginning of August 2014. Occupant 2, who commuted to work each weekday throughout the period can be seen to leave the dwelling at a regular time each morning and similarly return each evening. Occupant 1 (the author), spends most of their time indoors!

**Algorithm 1** Occupancy detection algorithm

```
1:  device1 ← 00 : 00 : 00 : 00 : 01                ▷ Store MAC address of device 1
2:  device2 ← 00 : 00 : 00 : 00 : 02                ▷ Store MAC address of device 2
3:  while True do
4:      time ← systemTime
5:      for device in [device1, device2] do
6:          if inRange(device) then                  ▷ Check if device responds to poll
7:              writeToFile(time, device, PRESENT)
8:          else
9:              writeToFile(time, device, AWAY)
10:         end if
11:     end for
12:     wait(10)                                    ▷ Wait 10 seconds before polling again
13: end while
```

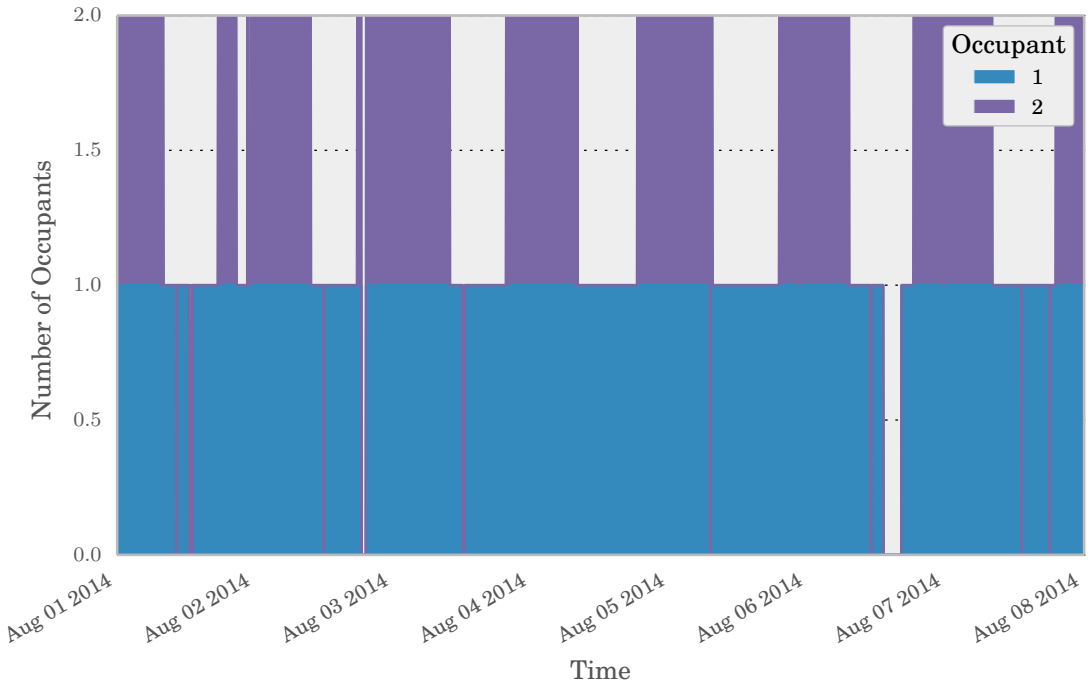



**Figure 2.4:** Number of Occupants for Period 08-01-2014 00:00 - 08-08-2014 00:00

## 2.4 Electrical Monitoring

Monitoring of the electrical supply in the home was accomplished at an aggregate 'supply' level and at an appliance level, with intrusive monitors placed on a significant number of appliances in the home.

Kelly and Knottenbelt's **rfm_edf_ecomanager** software [36] was flashed on a Nanode RF: an open source micro-controller board which has Ethernet connectivity and a 433mhz wireless module [43]. The software enables the Nanode RF to interface with EDF EcoManager Transmitter Plugs, and various Current Cost sensors/transmitters. The NanodeRF is supplied with a Helical antenna, which has a smaller aperture than the quarter-wave whip antennas used in [36]. However, testing showed that the device was capable of communicating effectively with transmitters placed at the extreme edges of the dwelling, and, due to the sub-microwave frequency even through several concrete walls to a transmitter placed at the electricity meter.

**Figure 2.5:** An EDF EcoManager Transmitter Plug

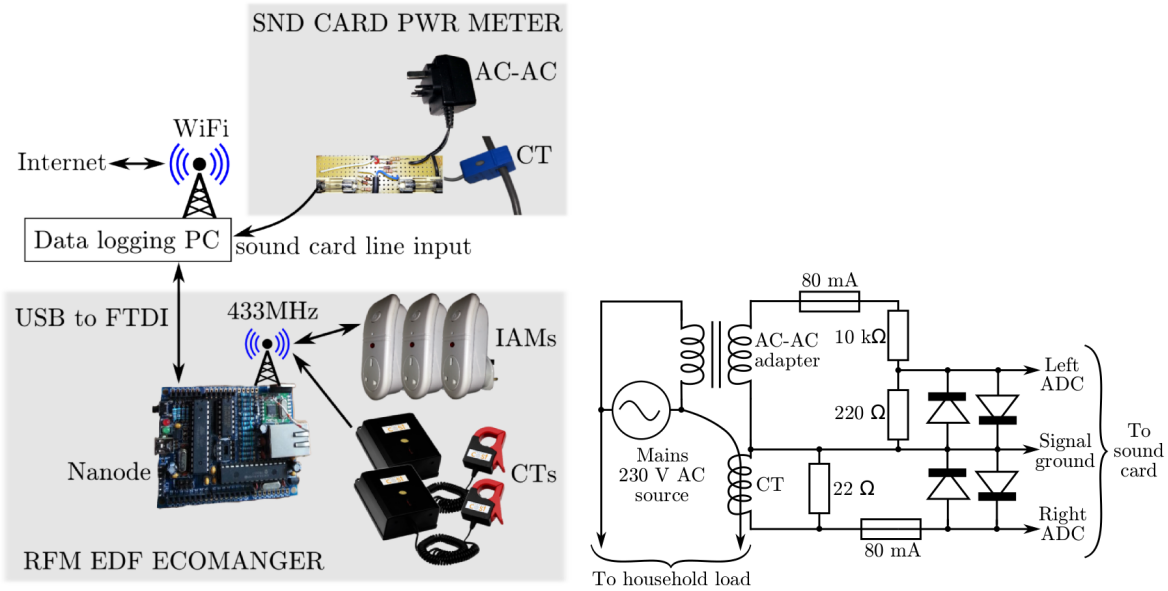
EDF EcoManager Transmitters were deployed throughout the home on devices connected via plug-sockets to the mains supply. Current Cost Sensable Transmitters with 12mm current transformer (CT) clamps were used to monitor the power consumption of hard-wired kitchen appliances for which no plug-socket was used: Figure 2.6 shows this arrangement. The EcoManager Transmitter Plugs transmit in response to a poll every 6 seconds from the Nanode RF. The Sensable Transmitters broadcast data every 6 $\pm$ 0.3 seconds. Since they do not wait for a poll from the NanodeRF, collisions and packet loss are inevitable. The **rfm_edf_ecomanager** software running on the NanodeRF attempts to avoid collisions as much as possible by learning the transmission window of the Sensable Transmitters and avoiding transmitting for a short window before and after this interval.

The NanodeRF is connected to a low-power Atom PC, running Ubuntu Server 14.04. This PC runs a suite of 5 software packages **rfm_edf_ecomanager, rfm_ecomanager_logger, powerstats, babysitter, snd_card_power_meter**. These packages are described in detail in [36]. They allow for the NanodeRF to communicate with the various transmitters in the household and store this data to disk in the same CSV file format as MIT's REDD dataset [37]. The Atom PC also uses a USB Sound Card connected to a 'Sound Card Power Meter' (SCPM) designed and kindly provided by Jack Kelly. Since the USB Sound card was not capable of recording at a bit depth of 24-bits, modifications were made to the **snd_card_power_meter** software to enable recording at a bit depth of 16-bits. The software records the signal as an uncompressed 44kHz WAV file and periodically converts it to the lossless FLAC file format to save disk space, and saves a CSV containing 1-second downsampled active and apparent power to disk. The UNIX application rsync was used to perform a daily back-up of the data to a RAID array.

The Sound Card Power Meter consists of a standard AC-AC adapter plugged into a wall socket, which tracks the mains input linearly over a range of 185.5 V to 253 V [36], this is connected to a voltage divider circuit whose output is fed to one channel of the USB sound card's audio input. A CT clamp is placed around the mains supply cable and connected across a 22 Ohm burden resistor. The output is fed into the other input channel of the sound card. Figure 2.7, reproduced from [36], shows the circuit diagram and a system context diagram of the monitoring hardware.

**Figure 2.6:** CT clamps installed on individual kitchen appliance power cables

The SCPM was calibrated using a 'Watts up? PRO meter' [31] and a kettle. The Watts up? meter provides the ground truth voltage and current, and the kettle provides a purely resistive device with which to calibrate the amplitude of the induced current and voltage supplied by the AC-AC adapter to the SCPM, as well as the phase shift introduced by the sensor arrangement. A purely resistive device is desired as it allows for calibration with a reactive power of zero.

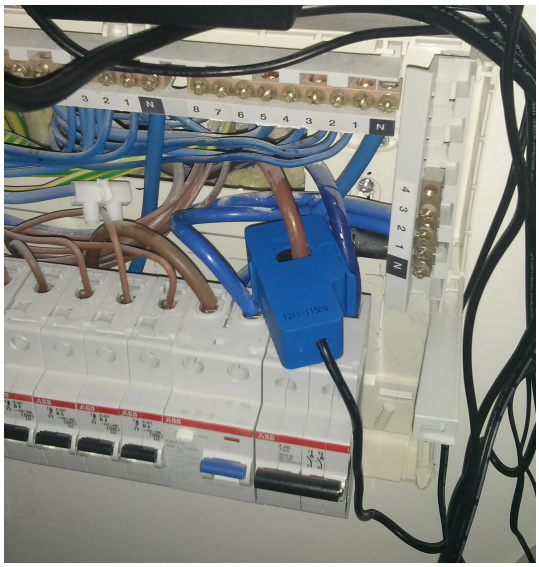



(A) System diagram.

(B) Sound card power meter circuit diagram.

**Figure 2.7:** '(A) shows the system diagram and the three major components of the system: 1) the data logging PC; 2) the sound card power meter (which uses the sound card on the data logging PC to record the output from an AC-AC adaptor and a current transformer (CT)) and 3) the RFM EDF ecomanager which uses a Nanode to communicate over the air with a set of individual appliance monitors (IAMs) and current transformer (CT) sensors. (B) shows the circuit diagram for interfacing a sound card to a CT clamp and AC-AC adaptor to measure mains current and voltage, respectively.' (Kelly and Knottenbelt, 2014: p.3)

**(a)** Atom PC with Nanode RF attached.



**(b)** CT Clamp in place within Consumer Unit.

**Figure 2.8:** Mains power supply monitoring equipment

### 2.4.1 Mis-calibration of Sensor Offset Angle

It became apparent after several weeks of monitoring that the offset angle introduced by the SCPM had not been detected correctly (despite multiple calibration efforts). Therefore, there is an offset in the reactive-power measurements computed from the recorded data of approximately 5.44 degrees in the signature space. This results in an upward tilt of the samples in the signature-space that can be seen in Figure 4.2 in Section 4. This should not affect the clustering and disaggregation performance since it is equivalent to a rotation of the samples about the origin in the signature space. It does result in the reactive-power being misrepresented in the plots however.

It is possible to correct for this effect but it was decided to continue collecting data as a longer period of consistently recorded samples would be preferable to having to scrap data and result in less information over-all.

### 2.4.2 Time Synchronisation

In order to align all data for future analysis, it was necessary to synchronise the clocks on the various devices used to collect data. The manufacturer of the temperature monitors confirmed that they updated their internal clock against a time server every time they connected to log data. I was therefore satisfied that the clocks on these would be accurate, as they were set to connect and log data at five minute intervals. The server logging occupancy was set up to synchronise with internet time-servers on a ongoing basis (`ntpd` on Linux monitors offset and jitter at very regular intervals and corrects the PC clock). The occupancy server was also then to broadcast time within the home network, and the power-logging machine set to use this as its 'master' time server. This allowed the electricity-logging machine and the occupancy logger to maintain synchronisation.

### 2.4.3 Summary

In this chapter we have seen that temperature, occupancy and power monitoring equipment was set up to monitor the author's home. All of the monitoring equipment had some tendency to 'drop' samples. Each of the monitors sampled at a different rate, which must be corrected for when performing analysis on this data. Time was spent attempting to hijack the data transmissions of the wireless temperature sensors, but due to the manufacturers efforts to protect their proprietary protocol, this did not prove possible. Efforts were made to ensure that the time would be synchronised across monitoring devices in order to enable later data alignment.

# Chapter 3

# Power Normalisation and Edge Detection

## 3.1 Introduction

Once the sensors had been installed in the household, attention was turned to implementing a disaggregation system based upon steady-state detection. The system to be built was divided into a pipeline of operations, shown in Figure 3.1. The pipeline begins with the parsing of raw sensor data, which is then normalised. Once the power-measurements have been normalised, edge-detection is performed to extract the start and end of steady-state periods of power consumption. The edges extracted from the normalised data are then passed into a pairing procedure, which matches ON and OFF transitions into pairs. Once the pairs are formed, the magnitude of each OFF and ON transition in a pair is averaged, and the power-state formed passed into a clustering algorithm. The final stage of the process is matching the clustered power-states to real-world appliances.

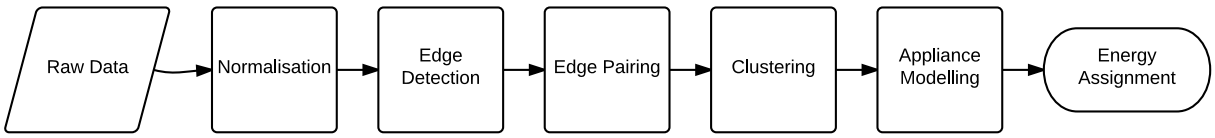



**Figure 3.1:** The disaggregation pipeline.

Wherever possible, code was implemented to allow for 'dynamic' disaggregation: that is to say disaggregation of a 'live' signal as it arrived, rather than purely on a pre-recorded dataset. This decision was made as future extensions to the work would allow for real-time feedback to be provided to a householder of their power consumption, and devices that were active at a particular moment. Compromises in execution speed had to be made as a result of this decision. `numpy` allows for very efficient 'broadcasting' vector operations to be made on arrays of data, where the array size is pre-defined and in most cases immutable. Optimised, pre-compiled C code can be used 'behind-the-hood' in such cases. As every attempt was made to allow for a live disaggregation, in many cases, these broadcasting operations were not used, and an iterative approach had to be used instead, with significant performance impact. However, it is still possible to process multiple months of data for disaggregation in a few minutes on a modern PC, which means that real-time operation of the algorithm is easily possible.

The processing pipeline for data measurements begins with the data being saved to a CSV file that can be loaded into memory within a Python Shell. CSV files are read and parsed directly into `pandas` DataFrame objects wherever possible. These objects are fast to perform aggregation, grouping and iteration operations upon. More importantly, they feature powerful tools for the manipulation of time-series data, which is extremely important in the context of this project. Functions can be applied to rows, columns or cells within a DataFrame, and the entire frame

iterated to extract information line-by-line. The line-by-line extraction was used whenever passing data into the various components of the disaggregation framework, as this best simulates the arrival of data in real-time .

## 3.2 Normalisation and Data Preparation

The first stage in the disaggregation algorithm involves the detection of 'Edges' within the aggregate household signal. An Edge is the transition between two constant power states, which should indicate the switching on or off, or change of state, of a device.

The use of Edges in power states as input for the algorithm is desirable since states are additive when two happen coincidentally. This, in theory, allows for the decomposition of compound state changes to be decomposed into their component parts, given enough observations of their individual states [27]. Edges are also visible at various sampling rates, which is not a property that necessarily applies to transient phenomena, and certainly does not apply to harmonics.

One complication of using Edges in the power state of a household is that the supplied Voltage, $V$ to the home is not actually constant but is allowed to vary in the UK by $+10\%$ to $-6\%$ [36]. Since a linear device connected to this supply will also vary in the current it draws by up to $\pm 10\%$ the actual power consumption may vary by over $\pm 20\%$. Table 3.1 shows small Voltage variations on a secondly basis, and a larger voltage difference over a period of several weeks.

| Time | Active Power / Watts | Apparent Power / VA | Voltage / V |
|---|---|---|---|
| 2014-06-30 10:09:23.500000 | 489.17 | 682.83 | 244.46 |
| 2014-06-30 10:09:24.500000 | 498.1 | 690.4 | 244.51 |
| 2014-06-30 10:09:25.500000 | 514.22 | 705.38 | 244.57 |
| 2014-07-27 12:31:58.300000 | 454.71 | 654.2 | 246.29 |
| 2014-07-27 12:31:59.300000 | 452.84 | 653.08 | 245.99 |
| 2014-07-27 12:32:00.300000 | 446.84 | 648.67 | 246.16 |

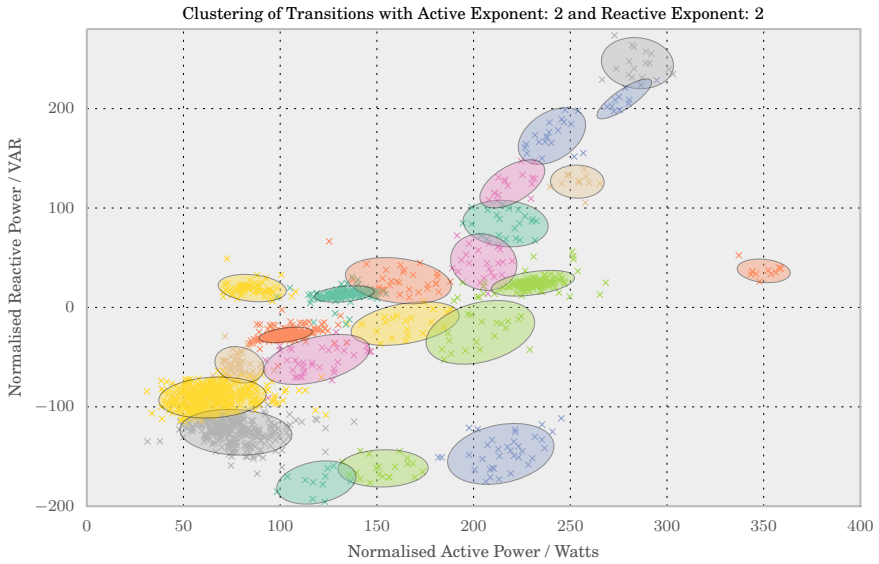
**Table 3.1:** Voltage difference across a few seconds and a 28 day period

In order to reduce the impact on the signature space of an appliance (i.e, the variation in its characteristic power states) the power measured at the mains by the Sound Card Power Meter is 'Normalised'. We define Normalised Power, $P_{Norm}(t)$ - where $P(t)$ and $V(t)$ are Power and Voltage respectively - as follows:

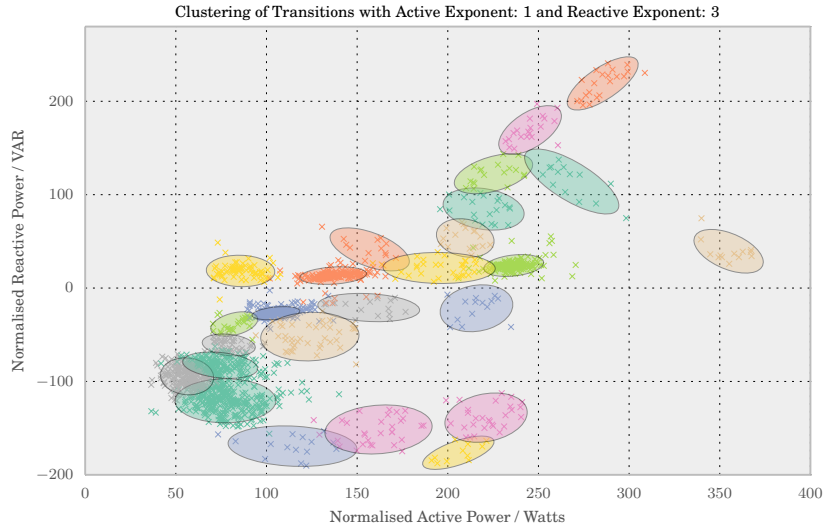
$$P_{Norm}(t) = \left(\frac{240}{V(t)}\right)^2 P(t) \tag{3.1}$$

Normalised Power can be thought of as what the power *would be* if the utility provided a steady 240V supply to the home, and if loads obeyed a linear model. It is a more consistent signature than power, and hence minimises the signature space of appliances, which will be important when clustering state representations later [27].

There is an interesting complication to this normalisation, which is that the optimal normalising exponent for non-linear devices is not necessarily 2. In [27] Hart notes that 'stabilised' resistive devices, such as a kettle, keep their resistance constant as a result of the water stabilising the temperature of their heating element. However, something like an incandescent light-bulb, whose power consumption increases more slowly than quadratically, is better modelled with a normalising exponent for its Real power of 1.5. Since a differing assortment of devices are present within any two homes, an exponent of 2 is chosen as a compromise, in the hope it will enable the discrimination of the maximum number of appliances. Figure 3.2 on page 34 shows a comparison of the clustering achieved in the crowded region between 0 and 400 Watts active power, where more than 10 state-transitions were grouped in each cluster. Cluster boundaries are drawn at 2 standard deviations from the multi-variate distribution center. In Figure 3.2a both Reactive and Active Power exponents are set to 2; in Figure 3.2b the Reactive exponent is set to 3, and Active set to 1.




**(a)** Clustering with Quadratic Exponents




**(b)** Clustering with Non-Quadratic Exponents

**Figure 3.2:** Comparison of clustering of power states over the period 2014-06-30 - 2014-07-27 with quadratic and non-quadratic exponents of normalisation

It is easy to see that the clustering is affected significantly, with the uppermost-right cluster pushed away from the nearby clusters in the first example, and compacted into them in the second.

Once the raw power measurements are loaded into a `pandas` DataFrame, in the format seen in Figure 3.1, it is necessary to compute the reactive power at each moment in time. The relationship between the real (active) power, $P$, reactive power, $Q$, apparent power, $S$, and phase-angle, $\phi$, in an AC circuit is described by the 'Power Triangle'. This is shown in Figure 1.15.

Thus:

$$S^2 = P^2 + Q^2 \tag{3.2}$$

and:

$$Q = \sqrt{S^2 - P^2} \tag{3.3}$$

Our monitoring equipment collects data on apparent and active power, it is thus possible to compute the reactive power at any moment in time by substituting in our known power parameters and solving Equation (3.3).

Having computed the reactive power, we normalise both the active and reactive power using Equation (3.1). The prepared data are then ready for Edges to be detected.

We encapsulate the functions that are applied to the raw data into a DataPrep.py Python module, so that it can easily be imported into any script that requires the functions. This package also includes functions to import homes from the UK-DALE dataset.

## 3.3 Edge Detection

Edge detection is performed in the manner described in Hart, 1985 (p27-28) [25], but in a dynamic fashion which does not require the storage of all previous measurements in order to compute the changes in state. Edges are output by the algorithm as a state is 'exited'. This means that the algorithm lags the current active state at any one time. This is necessary for averaging to be performed on the power state before the Edge is output. Algorithm 2 describes the Edge Detection.

The section in Algorithm 2 between lines 20 and 30 enables the Edge detection algorithm to 'ride over' transients in the signal. We do **not** record a transition until the boolean $C$ (which is set to True if the power has been fluctuating more than 15W for more than one sample) is False, but the boolean $I$, which indicates an instantaneous change, is True. This is what enables the algorithm to avoid classifying transients, incorrectly, as prolonged steady states, but it does mean that the algorithm only records the state **after** that state has finished.

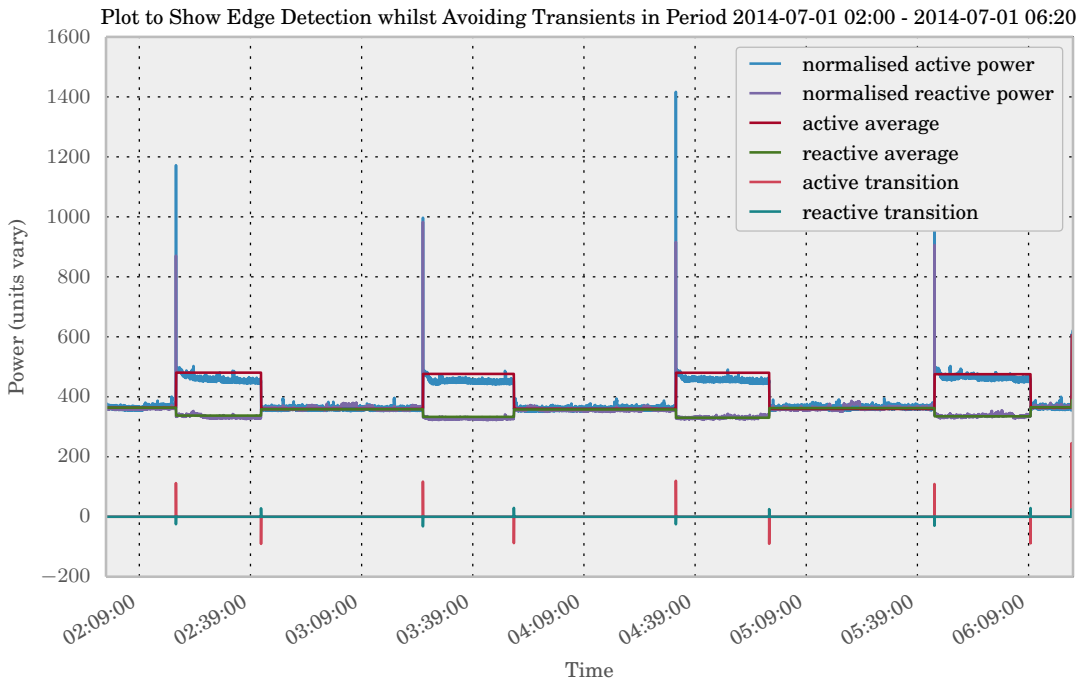



**Figure 3.3:** Plot showing transients caused by refrigerator compressor start-up, and the edge detection algorithm transient avoidance and averaging effect.

Let us consider the performance of the algorithm in classifying cycles of the fridge-freezer in the author's home.

Figure 3.3 shows a period through the early hours of the morning on 01/07/2014. During this period, both occupants are present in the home and fast asleep. Aside from the baseline load (around 400W, and mostly attributable to computer servers running permanently!) the only device

**Algorithm 2** Edge Detection Algorithm

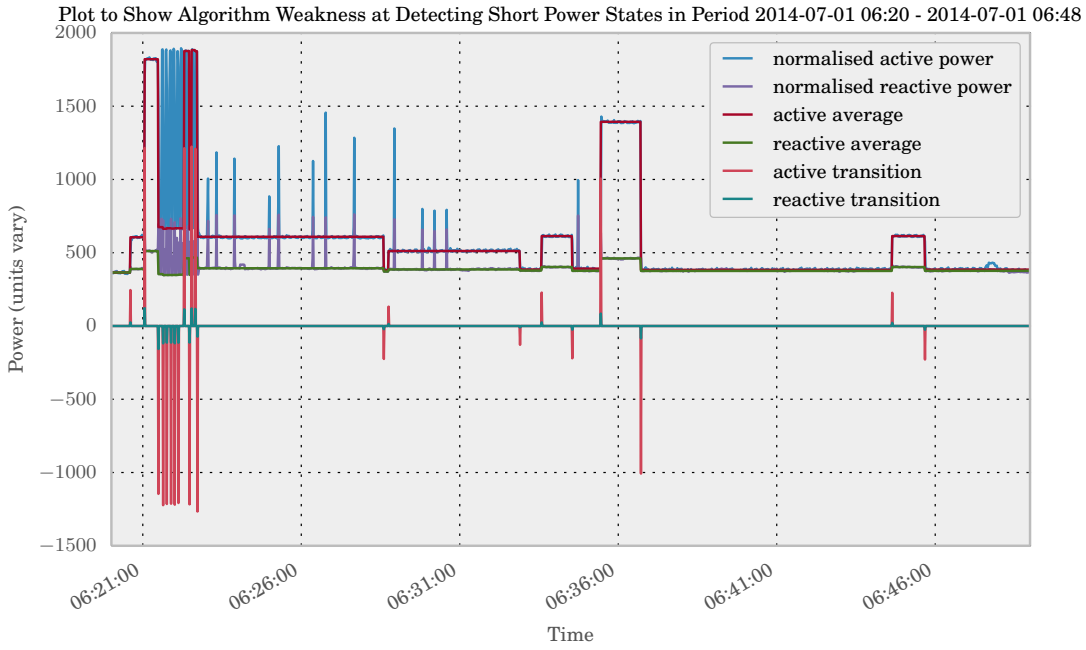
```
1:  estimate ← (0, 0)                          ▷ Estimate of actual power level during steady period
2:  last ← (0, 0)                              ▷ Last power value that was steady for at least two measurements
3:  power ← (0, 0)                             ▷ The measurement from the previous second
4:  I ← False                                  ▷ Boolean indicating if power level is changing this instant
5:  C ← False                                  ▷ Boolean indicating if change is in progress over multiple seconds
6:  N ← 0                                      ▷ Number of measurements in a state
7:
8:  threshold ← 15                             ▷ Threshold power variation for steady state
9:  noiseLevel ← 70                            ▷ Noise level, in Watts, below which transition is ignored
10: minSamples ← 2                             ▷ Minimum number of samples in valid state
11: time ← startTime
12: transitions, steadyStates ← [ ]            ▷ Empty lists for transitions and states detected

13: for M ∈ Measurements do                    ▷ M is a 3-tuple containing active, reactive power and time
14:     change ← (M − power)

15:     if any(change) > threshold then
16:         I ← True
17:     else
18:         I ← False
19:     end if

20:     if I and not C then
21:         lastTransition ← (estimate − last)
22:         if any( abs(lastTransition) ) > noiseLevel then
23:             if N > minSamples then
24:                 transitions.append(lastTransition, time)
25:                 steadyStates.append(estimate, time)
26:             end if
27:         end if
28:         last ← estimate
29:         time ← M.time()
30:     end if

31:     if I then
32:         N ← 0
33:     end if

34:     estimate ← ((N × estimate) + M)/(N + 1)
35:     N ← N + 1
36:     C ← I
37:     power ← M
38: end for
39: return transitions, steadyStates
```

that cycles states throughout the period is the fridge-freezer. The state change is characterised by a large start-up transient of up to 1000W of active power. One can see from the chart that the state transitions (Edges) recorded by the algorithm are of an amplitude that is the difference between the **averages** of each sequential state. During the period when the compressor is active, this average **includes** the energy used in the transient period, but extracts a signature for the device that is simpler and more closely reflects the actual steady-state behaviour of the fridge-freezer.

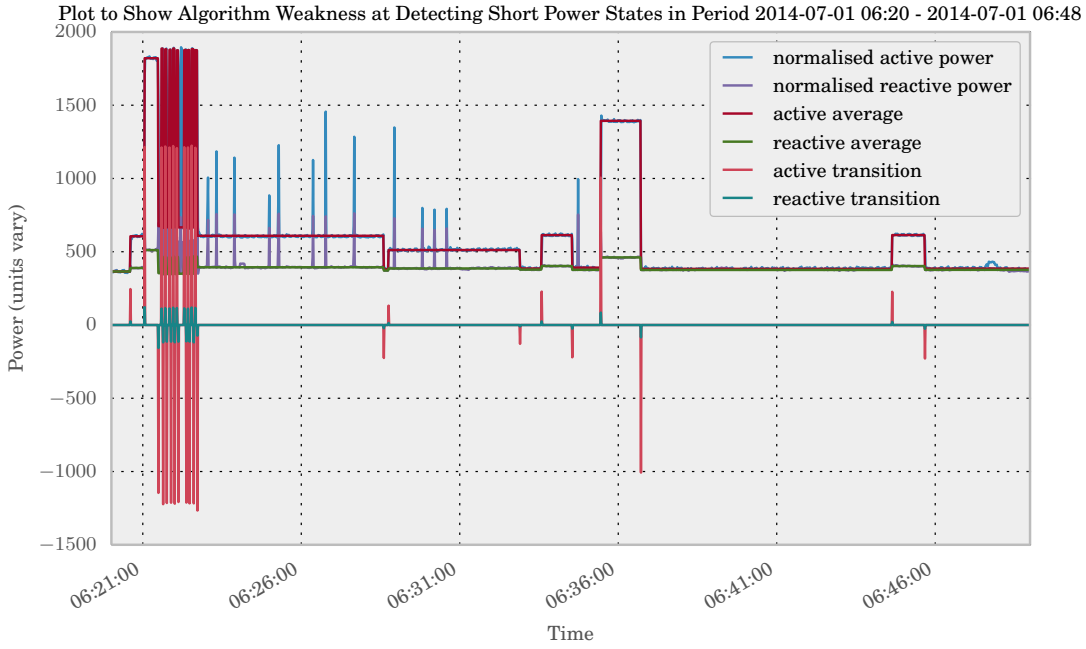
One can also see in this plot the noise that is present throughout the period under inspection.

The averaging of power during steady states provides for a cleaner extraction of features, with the vagaries of noise on the supply cancelled to a significant degree.

Another property of the algorithm as described, is that it could be easily modified to record the presence of a transient at the start of a state, and perhaps properties such as its slope and peak. This may provide further insight into the characteristic signature of certain appliances, such as the compressors in refrigeration units.




**(a)** Plot showing failure of Edge Detection Algorithm to detect very short lived states with a state length filter of 2 samples.




**(b)** Plot showing improvement in short lived state detection with no state length filter.

**Figure 3.4:** Comparison of Edge Detection Performance with a 2 second state length filter and no state length filter.

Figure 3.4a shows the weakness of this Edge Detector at detecting short lived power states. The Figure shows a 28 minute period in the morning, where a resident gets up, switches on various lights, and activates the household Nespresso coffee maker. The coffee machine contains a heating element that cycles rapidly after the water in the reservoir reaches the target temperature. The on period of this cycle is typically around 1 second. This poses issues for the Edge detection algorithm, as it does not consider one measurement to be a steady-state, and thus does not record the 'ON' transition, though it does record the 'OFF' transition. This does not cause us a cascading problem in our disaggregation, as our transition matching algorithm to be discussed in the next section will gradually discard unmatched negative transitions. However, it does mean that the power consumption of the coffee maker is likely to be underestimated by a considerable factor.

If we decrease the number of samples to consider a steady-state, or eliminate this filtering consideration entirely, as in Figure 3.4b we can detect the short lived states of the coffee maker. It is possible to see in this figure that the 'ON' transitions into the short lived heating states are correctly flagged. These are visible as red spikes that are not present in Figure 3.4a Whilst this works in this instance and at first glance appears to remedy the problem, it will cause noise spikes and other transient phenomena to be output from the algorithm as steady-states. This increase in noise is not desirable, as it will make clustering valid steady-states more difficult.

This discussion will apply to any device where a state exists that is approximately equal to our sampling rate. Increasing the sampling rate would allow us to account more accurately for device states, as a greater number of samples will exist in the state to be analysed.

As discussed by Hart in [25] a compromise must be struck between the desire to accurately account for the energy consumed by devices and the undesirable consequence of erroneously including noise in our list of state transitions.

In concluding our discussion of the Edge Detection Algorithm, it is worth noting that the **aim** of this algorithm is to detect **steady-states** and as such, whilst it may be appealing to include short lived phenomena such as those of the heating element cycling in the coffee-maker, it is debatable as to whether a 1-2 second activation time can truly be considered a steady-state when our sampling-rate is of this order. In this report, we will not consider such short duration phenomena as steady-states when they are of the same order as our sampling rate.

## 3.4 Pairing Transitions (Edges) to Capture States

After Edge detection has been performed, we must process any transitions highlighted as state changes and pair together transitions that indicate the start and end of power-states.

The PairBuffer implements the logic described in Hart 1985 (p33) [25]. Once again, this is implemented in a fashion that allows for the dynamic execution of the disaggregation process. Thus, the code that provides for the pairing of transitions is encapsulated in an independent class from the Edge detection algorithm. The class is implemented as a 'buffer' for transitions passed from the Edge detection process. We refer to this class as a PairBuffer. Transitions are passed into the PairBuffer in the order in which they occur. Thus, the oldest transitions are added to the PairBuffer first. Transitions are passed to the PairBuffer as 3-tuples, containing the active power, reactive power and a DateTime reflecting when the transition occurred.

An instance of the PairBuffer is instantiated with certain tunable parameters, the defaults for which are set as per the description in Hart's 1985 work:

- **bufferSize**
  We define the **bufferSize** as the number of transitions that can be stored in the buffer before the oldest transition is removed and discarded. The **bufferSize** is important, as it prevents old, unmatched transitions from being matched with a transition that potentially occurs many hours or days later. It is possible and probable that on occasion the transition that defines the start of a new state will not be detected, but that the transition that marks the end of it will. The converse also applies. This could occur due to the simultaneous switching of two devices or a noise inducing event causing the transition to be missed or misclassified in magnitude. When transitions are missed, we do not wish to pair transitions inappropriately over very

long time scales, so the gradual removal of old items from the PairBuffer is desirable. A large bufferSize will tend to allow the pairing of transitions belonging to states that last a long period of time, a small bufferSize may delete entries too rapidly, and fail to pair transitions that belong to valid, long-lived states.

Set to a default value of 20 transitions in the Prototype.

- **maxTolerance**
  We define the **maxTolerance** as the maximum difference of Watts or VAR between both the active and reactive components of two transitions if they are to be paired as belonging to the same state. The **maxTolerance** is used for transitions for which the magnitude of both active and reactive components is less than a **largeTransition** figure, at which point a percentage based tolerance is used.

  Set to a default value of 35 W/VAR in the Prototype.

- **largeTransition**
  We define the **largeTransition** figure as the number of Watts or VAR at which we allow the matching tolerance to be determined on a percentage basis, rather than an absolute allowable tolerance. This is based upon the expectation that all classes of devices will produce transitions that fit a multi-variate Gaussian pattern, and that the variation in transition amplitude will be larger on an absolute basis for large transitions.

  Set to a default value of 1000 W/VAR in the prototype.

- **percentTolerance**
  We define the **percentTolerance** as the percentage tolerance to use to determine whether two transitions are an allowable pair, once a transition's magnitude in either active or reactive power exceeds the **largeTransition** threshold.

  Set to a default value of 3.5% in the Prototype.

Figure C.1 in Appendix C shows the UML representation of the PairBuffer class, which implements a specialisation of the Python Deque class as the transitionList into which unpaired transitions are inserted. The standard Deque class can be viewed as an efficient List which has a maximum length property. Above this length any new items to be added cause the earliest inserted items to be deleted from the List. This functionality matches our desire to implement a fixed length buffer.

The standard Python Deque implements efficient 'pop' operations from the left and right of the list, but does not implement an arbitrary index pop operation. The simple specialisation that has been created, MyDeque, implements a **popmiddle** operation that allows for the element at any index to be removed from the Deque. This is used when cleaning the transitionList of paired transitions.

In operation, transitions are added to the PairBuffer one at a time, as each is added, an attempt is made to pair each transitions currently in the buffer. A valid pair of transactions is made if each of the following conditions are satisfied:

1. Both transitions have their *matched* flag set to False
2. The earlier transition has a positive active power component
3. When added together, both the active and re-active components of the resultant vector are less than the **maxTolerance** limit in the PairBuffer - or less than the **percentTolerance** multiplied by the largest component of either transition, if either of the transitions had a component whose amplitude was beyond the **largeTransition** limit.

Valid pairs of transactions have their *matched* flag set to True, but remain in the buffer for a short period.

Once the PairBuffer fills to its maximum size (as determined by the **bufferSize** parameter passed on instantiation), the *cleanBuffer()* method is called. *cleanBuffer()* passes through the transitions in the buffer and removes those that are marked as *matched*. If there are no matched transitions, the next transition to be added will cause the ejection of the oldest transition from the PairBuffer.

As discussed by Hart in [25] the optimum way to compare transitions for pairing is to gradually move through the buffer, first comparing adjacent transitions, then gradually increasing the distance between comparators. This approach ensures that we match those transitions that occur near to each other (in time) first (which reflects the reality of state changes of appliances within a household) before checking more distant elements, between which other transitions will have occurred.

Algorithm 3 has been implemented as the *pairTransitions()* method within the PairBuffer class, in order to allow us to pair transitions in the manner discussed above.

**Algorithm 3** Pair Matching Algorithm

```
buffer_length ← len(transition_list)
pair_matched ← False
matched_pairs ← []

if buffer_length < 2 then                          ▷ Return early if only one entry in buffer
    return pair_matched
end if

for e_distance in range(1, buffer_length) do       ▷ Gradually step distance between elements
    idx ← 0
    while idx < (t_length − 1) do
        comp_index ← idx + e_distance
        if comp_index < t_length then
            val ← transition_list[idx]
            if val.active_power > 0 and val.matched = False then
                comp_val ← transition_list[comp_index]
                if comp_val.matched = False then
                    v_sum ← (val + comp_val)
                    match_tols ← get_tols(val, comp_val)      ▷ Get tolerances for transition pair
                    if abs(v_sum) < match_tols then           ▷ Check for a matching pair
                        transition_list[idx] ← True
                        transition_list[compIndex] ← True
                        pair_matched ← True
                        matched_pairs.append([val, comp_val])
                    end if
                end if
            end if
            idx ← idx + 1                                     ▷ Step through buffer
        else
            break
        end if
    end while
end for

return pair_matched, matched_pairs                 ▷ Return whether we have found a pair
```

# Chapter 4

# Clustering Power States

## 4.1 The Need for Clustering State Transitions

Clustering of paired transitions is one of the most important aspects of the steady-state driven disaggregation approach. Ultimately, we are attempting to identify power-states of appliances that are located within the home under investigation, in order to do this, we attempt to group paired transitions into clusters that are representative of a single power-state of an appliance.

We will never observe precisely the same state transition amplitude for a single, real appliance power state. Consider that the following factors (which represent just a few of the many contributory items) will all, to a greater or less extent, affect the state transition computed by the Edge detection algorithm:

- **The Effects of Averaging**
  As explained in Section 3.3, an averaging process is in place which averages the individually sampled power values during a steady-state period (within which, power is allowed to vary within a window of by ± 15W or VAR (or other tolerance band). The length of period involved and minor fluctuations of power within it will result in minor differences in the transition value recorded.

- **Thermal Effects on Resistance**
  Electrical resistance in an ordinary conductor (that is to say, non-superconductor and at a temperature in a range that might reasonably be experienced in a domestic setting) is dependent on the likelihood of collision of electrons within the conductor. Temperature has an important contribution to this, as it increases, resistance in a conductor increases. The change in resistance as a result of temperature change is expressed as follows:

  $$R = R_0 \left(1 + \alpha(T - T_0)\right) \tag{4.1}$$

  Where $R$ is the resistance after a change in temperature from $T_0$ to $T$, $R_0$ is the resistance in the conductor at $T_0$ and $\alpha$ is the temperature co-efficient of resistance, in units of $K^{-1}$.

  Since:

  $$P = I^2 R \tag{4.2}$$

  Where $P$ is Power in Watts and $I$ is Current in Amps, changes in ambient temperature will have a very slight effect on the power consumption of devices.

  Appliances that make use of resistive elements for heat or light generation will be more affected still. A kettle, grill or oven element, and incandescent light bulb will exhibit a variance of temperature between hundreds or thousands of K.

- **Measurement Noise**
  Signal noise may be induced by various sources in our measuring apparatus. Noise from the components within the monitoring PC may affect the signal recorded by the sound card power meter. The sound card used will also introduce errors as a result of conversion via its Analogue-Digital Converter (ADC).

  Noise is also introduced in the system by resistive elements subject to heating. The thermal agitation of electrons in a circuit gives rise to Johnson-Nyquist noise [47].

Figure 4.1 shows a two minute period during which the active power and Voltage are contained within a reasonably narrow band, but significant noise can be seen in the signal. This is typical of the data being collected and the illustrative period was selected arbitrarily (except to ensure that it fell within what would be considered a steady-state period).

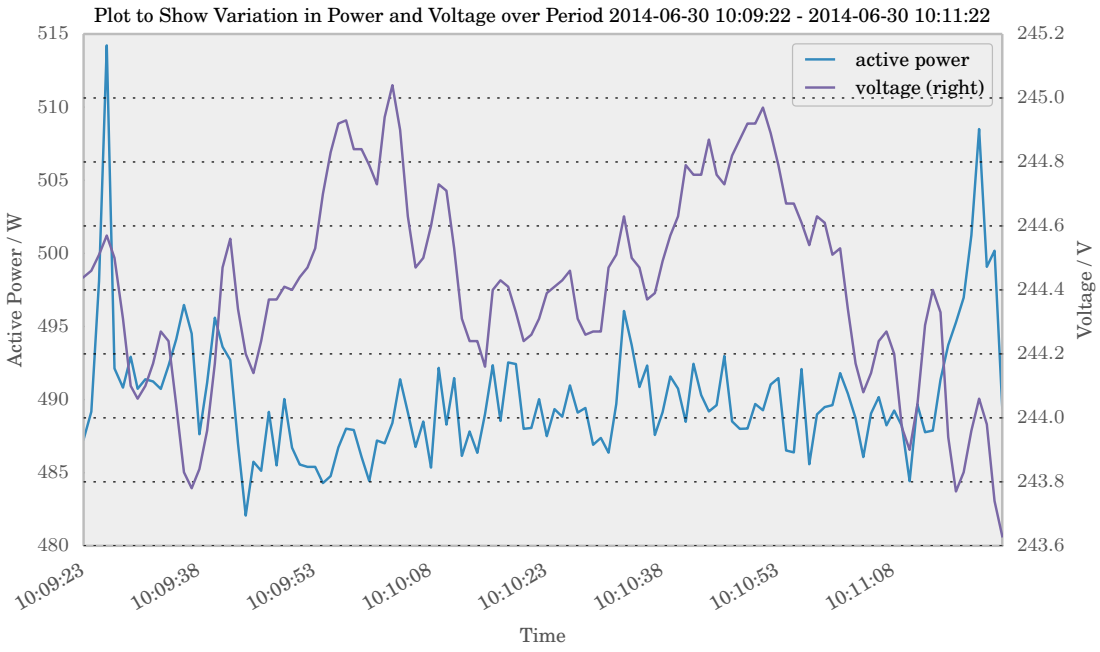



**Figure 4.1:** Illustrating Noise in Voltage and Power over a 2-minute Window

As a result of the natural dispersion of transition values for power-states, we must apply a 'clustering' process to identify those transitions that belong to the same appliance power-state, and distinguish each state from another.

## 4.2 Visualising the Data to be Clustered

In order to assess the suitability of our clustering algorithms, and to better understand the signature space over which we are performing clustering, it is useful to visualise the data output from edge detection and pairing process.

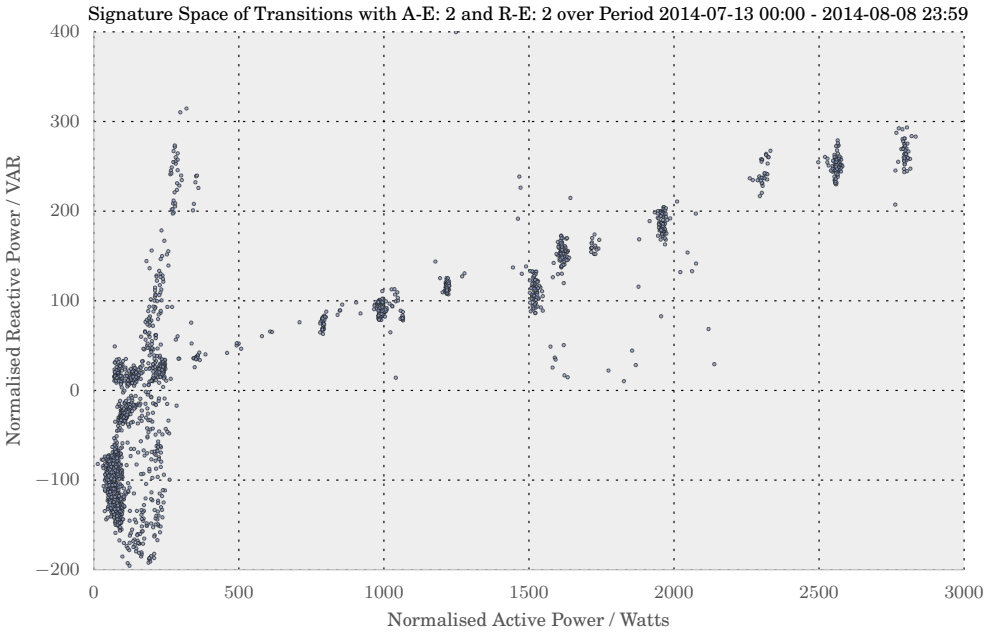



**Figure 4.2:** Averaged and Normalised Power States in Active/Reactive Power Signature Space

Figure 4.2 shows the signature space for the author's residence from data extracted in a period of slightly less than one calendar month. Clearly visible are structured agglomerations/clusters of points throughout the plane. One can obtain an intuition that clusters are more easy to distinguish between (at least upon visual inspection) at power-states of greater than approximately 500 Watts of active power. The clusters are more well separated in the signature space and their appears to be less 'noise'. In our case, noise might be considered as points, representing states, that do not correspond clearly with any of the well defined clusters.

Another consideration is that the clusters of points are not of the same density or shape. This is to be expected - some appliances are used more than others, and thus have more points contributing to a power-state cluster.

One can also distinguish that the region up to 500W reactive is very densely packed with points. Figure 4.3 shows a 'zoomed' view on this region. In this figure, dense clusters are visible, but there are also many points which do not seem to be strongly bound into clusters.

There is also a densely packed, but relatively large agglomeration of points in the very lower left of the figure, between 50 and 100W of active power.

It transpires that there are very many power-states produced within the household that fall within this region of the signature-space, but that do not belong to one single appliance. These include but are not limited to: televisions, computer monitors, amplifiers, computers and lighting circuits.

We may summarise some conclusions from inspecting this dataset and consider the means by which it was generated. These will be useful in our selection of a clustering algorithm:

1. **Clusters do not appear to be of equal density.**
2. **Clusters are not spherical, or all of similar shape.**
3. **Noise seems to be present in the dataset.**

4. **Even densely clustered points may not belong to one appliance's power-state.**

5. **We cannot know in advance the number of clusters that will be present in the dataset.**

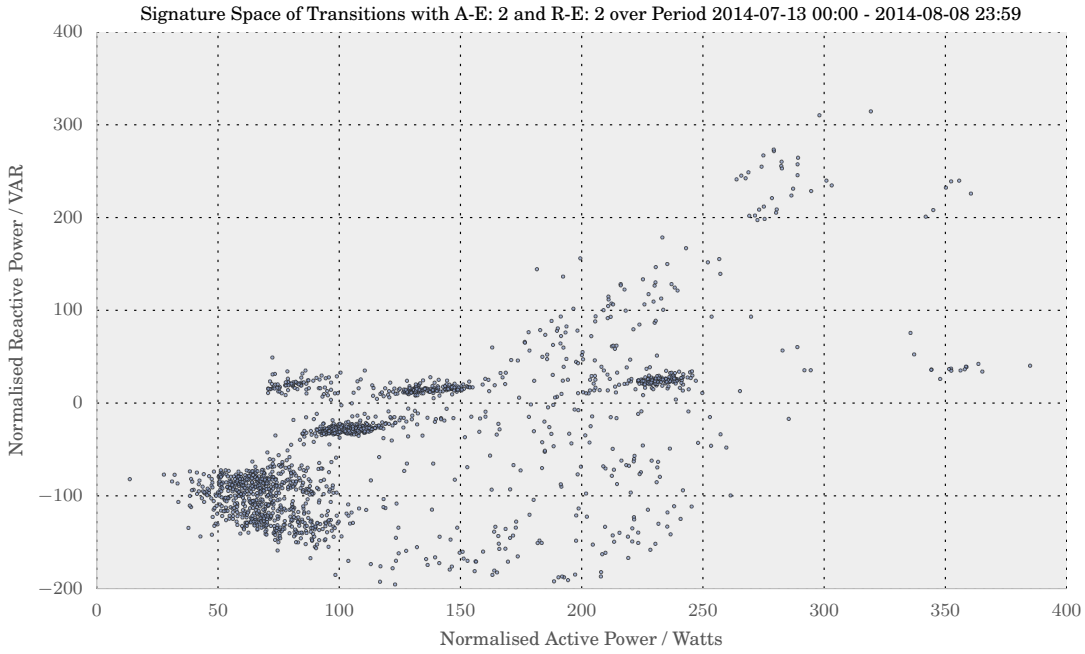



**Figure 4.3:** Averaged and Normalised Power States in Active/Reactive Power Signature Space up to 400W Active Power

## 4.3 Clustering Algorithms

Many algorithms exist to cluster data points in n-dimensional space into regions of high density, for which there appears to be local correlation between the data points. Our signature space is of two dimensions - active and reactive power. In [25], a signature space of 2.5 dimensions is used. The 'half' dimension in question is a flag to indicate on which 'leg' of the 120V power supply provided to US homes the device is connected - or, if a 240V device - if it is connected to both legs of the power supply. The work that has been implemented in this project does not currently enable for this extra information to be incorporated into its clustering process. This decision was made as the system was developed and tested on data collected in the UK, where domestic supply is single-phase 240V without the 'centre tap' three-wire deployment that is used in the U.S. It would be relatively straightforward to extend the existing classes to incorporate extra information in the disaggregation process, but this has not been implemented thus far.

Arguably, we might extend our signature space to three dimensions to include the duration of states. One argument for this might be that we have already discovered that the refrigerator in our dwelling cycles at very regular intervals throughout the day, and remains on and off for a similar length of time in each case. This might lead us to consider that we may discriminate the refrigerator from other devices also consuming 100-150W of active power by clustering together states that exhibit the characteristic duration. Whilst at first inspection this may seem to be the case for the refrigerator we should also consider the case when the refrigerator is loaded with a fresh batch of room temperature goods. In this situation, the refrigerator is likely to remain in its ON state for a longer duration than in a regular overnight period when the goods contained within are already cooled to a significant degree. The specific heat capacity of the goods within the refrigerator will also determine how much energy is expended in cooling them to the desired temperature. Thus, there are likely to be many causes of variation in the duration of the refrigerators ON cycle and we should not necessarily expect that these will be distributed in such a fashion as to improve the correlation between states that truly belong to the refrigerator and improve our clustering performance.

Having established that our signature space will be 2-dimensional we must select an algorithm to perform our cluster analysis. The goal of which will be to distinguish sets of points in the signature space that are more 'similar' to each other than they are to other points within the signature space.

Multiple algorithms were assessed for their clustering ability, applicability to the problem domain, and generalisability in parameter selection.

The lack of dynamism in many of the algorithms discussed below is undesirable, but since this project is an exploration of the viability of the technique of edge-detection and appliance identification, we may still gain insights into the viability of the *other* aspects of the disaggregation process and nature of our data by using such a clustering algorithm.

## 4.4 K-Means Clustering

The standard `k-means` clustering algorithm was proposed by Lloyd in 1982 [41] after an initial description by Hartigan and Wong [28] in 1979. The algorithm aims to partition $M$ points (of $N$ dimensions) into $K$ Clusters such that the within-cluster sum of squares relative to the cluster's mean is minimised. The algorithm seeks local optima, and the number of clusters ($K$) to be found in the data must be specified in advance.

The algorithm must be initialised by assigning $K$ 'seeds', which are points selected from the distribution of $M$ points at which to initialise cluster centroids. Clustering then proceeds by iteratively assigning each point to its nearest cluster center (where nearest is defined as the Euclidean distance between the point and centre) and then updating the cluster center to be the mean of all points assigned.

The algorithm continues until convergence, which is achieved when the assignments of points to clusters no longer change on further iterations.

As the algorithm is heuristic, there is no guarantee that the convergence achieved will be the

global optimum as only local optima are computed but these are dependent on the locations of the initial seeds.

Seeds can be selected randomly from rows in the input data, or an initialisation method known as '`k-means++`', first described by Arthur and Vassilvitskii in [1]. In [1], the authors find that the `k-means++` initialisation method yields much better clustering than a random initialisation, and that a random initialisation can lead to clusters being distributed toward the centre of the distribution.

The approach taken in `k-means++` is to attempt to spread out the seeds as much as possible when initialised. This is achieved by selecting a first seed location randomly, and then selecting the other $K-1$ clusters by checking the distance for each data point from the nearest seed that has already been set, and selecting a new point on which to seed a cluster with a probability proportional to the squared distance of the point from the nearest existing seed.

Our input set would seem likely to lead to the pathological outcome for a random initialisation, as we know that in the very high density agglomerations of points in our input data, there are a high concentration of the total number of points. Therefore, we shall use the k-means++ initialisation.

- **Advantages**
  - Implemented in `scikit-learn`, relatively straightforward to implement in our software.
  - Relatively fast (for small $K$), `scikit-learn` implementation has average time complexity $O(K.M.T)$ where $T$ is the number of iterations to run.
  - Few tune-able parameters. 'Set K and forget'!
- **Disadvantages**
  - Requires number of clusters, K, to be specified in advance.
  - Due to Euclidean distance metric, essentially models clusters as spherical.
  - Tends to produce clusters of even size [1].
  - No dynamic execution:clustering must be re-run if points are added.

### 4.4.1 K-Means Evaluation

It would seem that since the `k-means` algorithm requires the number of clusters to be specified in advance, it might be ruled out for our purposes. However, we are able to quantitatively assess the 'quality' of the clustering with different values of $K$ by using a Silhouette Coefficient after each attempt at clustering.

This approach requires us to iteratively increase the value of $K$ and perform clustering, and to score the clustering. This is computationally intensive and wasteful, but as we do not generally have prior knowledge of the number of appliance power states we are left with little choice. We may still make some informed decisions about the minimum number of appliances likely to be visible in the signature space of a home to increase the lower bound of K, but we would struggle to generalise the upper bound for the number of power-states that will be visible (i.e, above our noise threshold) in *any* home.

The Silhouette Coefficient for any sample, $i$, is computed as follows:

$$s(i) = \frac{b(i) - a(i)}{max\left\{a(i), b(i)\right\}} \tag{4.3}$$

Where $a(i)$ is the mean intra-cluster distance for the cluster to which the sample $i$ was assigned, and $b(i)$ is the distance to the centroid of the nearest cluster to the sample other than the one to which $i$ was assigned. This can be thought of as measuring the average dissimilarity of $i$ to all the other samples in its assigned cluster and comparing this figure to the average dissimilarity of $i$ to its next nearest cluster. Figure 4.4, reproduced from [51], provides a graphical illustration. $a(i)$ is the average of the lengths of the lines *within* $A$ and $b(i)$ the average of the lengths of the lines from $i$ into the samples within $B$.

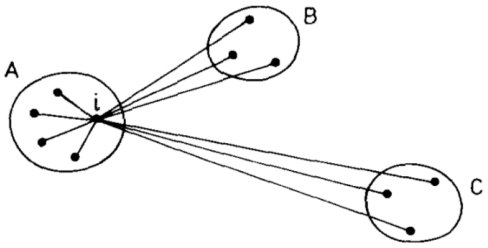



**Figure 4.4:** 'An illustration of the elements involved in the computation of $s(i)$ where the object $i$ belongs to cluster $A$' (P. Rousseeuw, 1987: p.55)

We can make use of the Mean Silhouette Coefficient (MSC) over our entire set of samples as a measure of the 'goodness' of fit of $K$ to the actual clusters in the dataset. We choose a minimum value of $K$ of 10. It is unlikely that in the general case any household will contain less than 10 distinct power-states above 70 Watts active power, and thus we commence iterating the value of K from this value and measuring the MSC for each value of $K$ up to a maximum. We choose a maximum of 30 as this should be comfortably in excess of the expected power-states for our data-sets and will give us some insight into the performance of `k-means` when it attempts to find a greater number of clusters in a dataset than the best fit.

Figure 4.5 illustrates the clusters formed by the `k-means++` algorithm when initialised with a value of $K$ between 10-30. We illustrate the clustering only over 6 values of K, such as to enable the reader to gain an understanding of how the clustering proceeds at high values of $K$. Of course, we can only draw qualitative conclusions from this visualisation, but it is apparent that clusters start to proliferate primarily in the lower range, whereas in Figure 4.3 we have already seen that this region seems to be noise filled, with rather few well defined (much less spherical!) clusters.

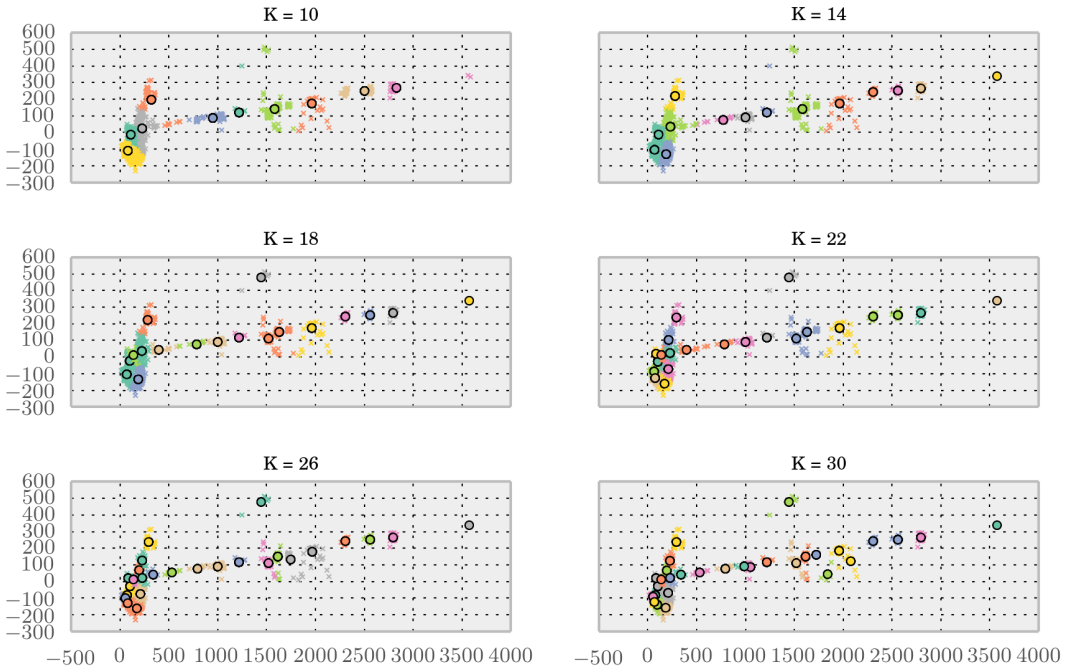



**Figure 4.5:** Visualising the outcome of K-means clustering in the range $K = \{10, ..., 30\}$. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid.

In Figure 4.6 we more closely inspect the region up to 400 Watts active power for the same clustering. We see the expected behaviour of `k-means` as $K$ increases: clusters develop of roughly equal spatial extent in spite of a lack of 'dense' agglomerations of points (except for the 5-6 visible in Figure 4.3.

In fact, if we perform silhouette scoring on the various clusterings for values of $K$ in the range 10-30, we find that the highest Mean Silhouette Coefficient is achieved with $K = 14$. Table 4.1 shows the MSCs for values of $K$. The table is presented in descending order of MSC.

It is apparent from these results that there is very little difference between the highest and lowest

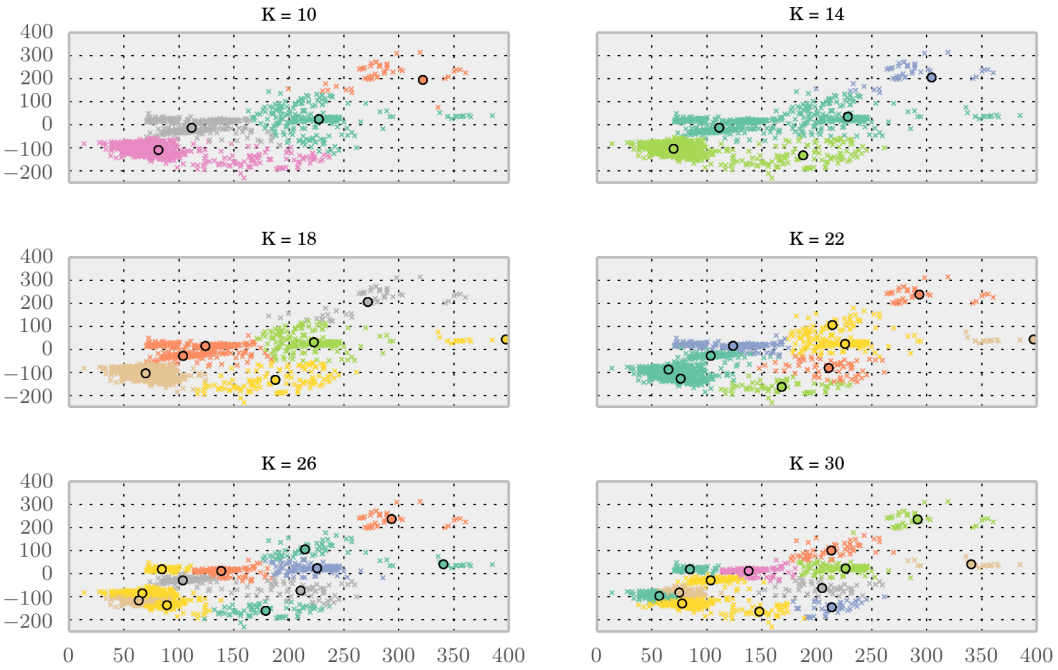



**Figure 4.6:** Visualising the outcome of K-means clustering in the range $K = \{10, ..., 30\}$. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid.

| K | Mean Silhouette Coefficient |
|---|---|
| 14 | 0.664668 |
| 13 | 0.662103 |
| 16 | 0.661994 |
| 27 | 0.659132 |
| 25 | 0.658403 |
| 15 | 0.658361 |
| 26 | 0.657185 |
| 19 | 0.652813 |
| 18 | 0.648121 |
| 17 | 0.637519 |
| 12 | 0.63377 |

| K | Mean Silhouette Coefficient |
|---|---|
| 24 | 0.63131 |
| 22 | 0.630858 |
| 28 | 0.629226 |
| 23 | 0.628705 |
| 21 | 0.625833 |
| 29 | 0.62449 |
| 11 | 0.623295 |
| 30 | 0.621711 |
| 20 | 0.618893 |
| 10 | 0.611739 |
| - | - |

**Table 4.1:** Mean Silhouette Coefficients (in descending order) for clustered data for varying values of K.

silhouette scores. It is also notable that if we consider the highest 6 results, all are extremely close in mean score, and there seem to be two ranges of K with similar levels of support: $13 \leq K \leq 16$ and $25 \leq K \leq 27$.

The maximisation of MSC at values of $K$ significantly lower than the number of power-states one might expect will have a significant impact on the success of disaggregation. Higher values of $K$ than the MSC metric would suggest actually enable more successful disaggregation. This is best explained by considering the ultimate goal of our clustering, and the nature of our samples.

We know that samples near to one-another in the signature space are *likely* to be the result of the same appliance. If one considers the plot for $K = 14$ in the upper-right corner of Figure 4.6, 5 clusters are visible. The centroids are also located inside the dense agglomerations of samples that were visible in Figure 4.3. This is a desirable result, however it is also evident (even at the low level of detail available in such a small graph) that the spatial extent of each cluster is very large, and the clusters include many points that are far from the dense central regions. As the distance of a sample from this central 'nucleus' of each cluster increases, the likelihood that the sample belongs to the same power-state as the samples packed into the nucleus decreases rapidly - this can be

understood in the context of our expectation that power-states will be modelled by multi-variate Gaussian distributions.

Thus, whilst the 5 clusters seem well supported by the data, the inclusion of samples far from the dense nuclei actually has a highly detrimental effect. It should be remembered that as well as the reactive and active power measurements visible in the signature space, **each sample also has associated with it a record of the start and end time of the power-state**. We will make use of the duration and time-of-use information of each power state in assigning clusters to appliances. Thus, the large clusters achieved with $K = 14$ have a highly detrimental effect on disaggregation performance. The higher values of K (indicated by the range $25 \leq K \leq 27$) provide for superior disaggregation ability.

We can gain intuition into why this should be the case: as $K$ increases, we gain clusters that were less well supported by areas of high sample density. *However*, as a result of our use of `k-means++` initialisation, and ensuring that our seeds are well distributed among the samples, we gain clusters of approximately equal size, of small spatial extent, that are well spread through the signature-space and contain points local to one another. Therefore, provided that these clusters are not so small as to have too few points with which to calculate useful metrics (such as time-of-use profile and a reasonable estimate of the duration of use) they should aid our disaggregation performance.

## 4.5 Mean Shift Clustering

The original mean-shift procedure was proposed by Fukunaga and Hostetler in 1975 [20] and was more recently shown to be of more general use in pattern recognition by Comaniciu and Meer in [13]. Mean shift clustering is a non-parametric technique to delineate arbitrarily sized clusters in a multi-modal feature space [13]. The only user-set parameter is a measure of the resolution of the analysis referred to as 'bandwidth'. Cheng showed in [9] that mean shift is a 'mode-seeking' process, which if used with a Gaussian kernel, makes it suitable for our application of finding regions of density within a 2-dimensional plane.

The algorithm can be considered as setting a window around a region, iteratively updating points in a given region - the Neighbourhood - rendering a centroid that is the mean of the samples within that centroids neighbourhood. Since the algorithm requires multiple nearest-neighbour searches during its execution it is not highly scalable.

One can consider the 'bandwidth' parameter in a flat kernel to be the radius of the neighbourhood within which points are clustered. This proves problematic for our problem domain. We know from inspection and our expectation that power-states in our signature space will be Gaussian multi-variate in distribution that at higher active power levels, points belonging to one power-state may be distributed over a larger space than at low power levels. In [14], Comaniciu, Ramesh and Meer propose a variable bandwidth mean shift procedure which may be more suitable to our data set, however, this has not been implemented due to time constraints.

The current `scikit-learn` implementation of the mean-shift procedure uses a flat kernel, which suffers the bandwidth issue described above by not modelling clusters in a Gaussian fashion. Therefore, the implementation used is from an upstream-branch of `scikit-learn` which introduces a Gaussian kernel and was written by Peter Jacob [33].

- **Advantages**
  - Implemented in `scikit-learn`, relatively straightforward to implement in our software.
  - 1 pseudo-parameter to set: the bandwidth.
  - Number of clusters does not have to be set in advance.
- **Disadvantages**
  - Does not scale to large N due to number of nearest-neighbour searches required.
  - Sensitive to selection of bandwidth, `scikit-learn` implementation uses fixed bandwidth.

– No dynamic execution: clustering must be re-run if points are added.
– Selection of kernel must be appropriate to dataset.

### 4.5.1 Mean Shift Evaluation

Again, we can make use of the Mean Silhouette Coefficient (MSC) to try and assess the goodness-of-fit of our clusters to the dataset. We make use of the bandwidth estimation function of `scikit-learn` to estimate bandwidths for various quantiles of the distribution of pairwise distances within the dataset. A quantile figure of 0.5 would represent a bandwidth returned equal to the median of all pairwise distances.

Table 4.2 shows the Bandwidth, MSC, and number of clusters obtained for varying quantile selections over our dataset.

Again, it is possible to see from this table that the Mean Silhouette Coefficient is highest when clustering results in very few clusters in the dataset. In the rightmost table, one can see that the MSC steadily declines until 25 clusters are formed - at a Quantile of 0.05. After this it rises sharply from 0.53 to 0.67 before declining again. This supports an improvement in cluster fit between 25 and 50 clusters.

Figure 4.7 on Page 51 shows the clusters formed for a bandwidth, $B$, in the range $130 > B > 15$. It is easily possible to see the explosion in the number of clusters formed as $B$ decreases. It is worth noting that the clustering of samples above 500W active power looks appropriate at a $B$ of 129.8244, and clusters in this range proliferate and become smaller as the bandwidth decreases, seemingly forming a poorer fit with the data.

| **Quantile** | **Bandwidth** | **MSC** | **# Clusters** |
|---|---|---|---|
| 0.3 | 452.99 | 0.822947 | 4 |
| 0.29 | 450.73 | 0.822947 | 4 |
| 0.28 | 444.37 | 0.822947 | 4 |
| 0.27 | 438.32 | 0.822947 | 4 |
| 0.26 | 412.24 | 0.827383 | 5 |
| 0.25 | 373.86 | 0.826495 | 5 |
| 0.24 | 340.33 | 0.826495 | 5 |
| 0.23 | 337.1 | 0.826495 | 5 |
| 0.22 | 333.21 | 0.826495 | 5 |
| 0.21 | 324.61 | 0.826495 | 5 |
| 0.2 | 294.66 | 0.834839 | 6 |
| 0.19 | 279.21 | 0.8347 | 6 |
| 0.18 | 266.96 | 0.8347 | 6 |
| 0.17 | 241.92 | 0.825602 | 8 |
| 0.16 | 229.86 | 0.825602 | 8 |

| **Quantile** | **Bandwidth** | **MSC** | **# Clusters** |
|---|---|---|---|
| 0.15 | 218.48 | 0.826372 | 9 |
| 0.14 | 205.42 | 0.83953 | 10 |
| 0.13 | 197.4 | 0.83953 | 10 |
| 0.12 | 192.86 | 0.83953 | 10 |
| 0.11 | 183.48 | 0.83953 | 10 |
| 0.1 | 165.64 | 0.820318 | 12 |
| 0.09 | 146.65 | 0.706691 | 13 |
| 0.08 | 129.82 | 0.690878 | 14 |
| 0.07 | 110.84 | 0.699352 | 14 |
| 0.06 | 87.55 | 0.586177 | 20 |
| 0.05 | 75.76 | 0.52944 | 25 |
| 0.04 | 65.78 | 0.671251 | 28 |
| 0.03 | 38.6 | 0.641272 | 47 |
| 0.02 | 28.78 | 0.643523 | 69 |
| 0.01 | 15.39 | 0.521995 | 134 |

**Table 4.2:** Mean Silhouette Coefficients for clustered data for varying values of Quantile/Bandwidth.

Unfortunately, as can be seen in Figure 4.8, in the lower active power range, the attribution of the dense agglomeration of samples that are visible are not fully accounted for by individual clusters until the Quantile is set to 0.02 and $B$ is 28.78. This comes at the cost of many smaller clusters being formed in the region by algorithm. However, as we saw with the `k-means` algorithm, provided that these have a reasonable number of points within them, it may not be detrimental to our disaggregation performance and we can perform post-processing on the clusters to filter those with few data points.

In fact, if the range for the Quantile, $0.02 \leq Q \leq 0.03$ is investigated more closely, we find that the MSC is maximised at a value of $Q = 2.9 \times 10^{-2}$ with Bandwidth $B = 36.64$ and upon inspection, all of the visible dense regions in the lower power range are identified with this value of Q. Figure B.1 in Appendix B shows this.

We will evaluate the success of mean shift clustering in Chapter 7 where the value of $Q$ is set to 0.029. It is worth noting that the time to optimise the $Q$ value by iteratively fitting the data-set

and computing the MSC is significant, we have to undertake many iterations and mean shift does not scale well. This is a barrier to the generalisability of mean shift clustering for our application.

Based upon the above analysis, it may be worth splitting our sample data into those samples with active power $< 500$W and those above this figure being clustered separately with higher bandwidth values. However, this reduces the generalisability of the clustering, as we cannot be sure that this level is sensible for all households.

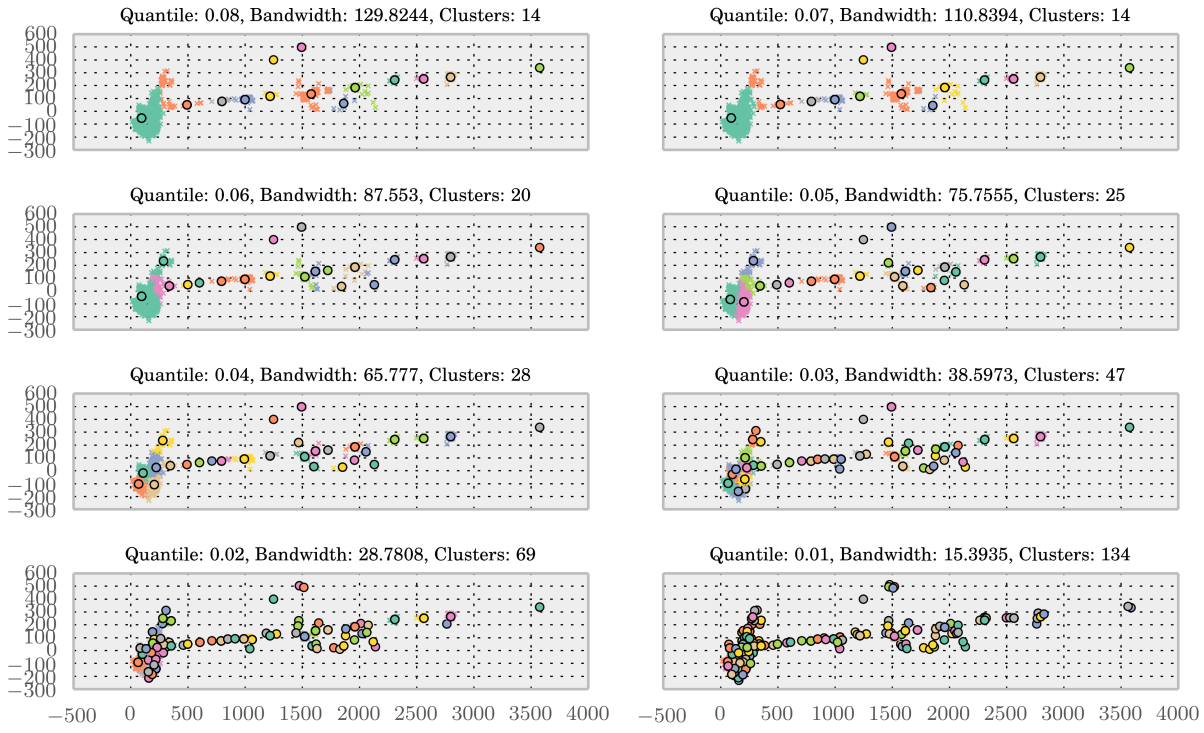



**Figure 4.7:** Visualising the outcome of mean shift clustering for varying Bandwidth values. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid.

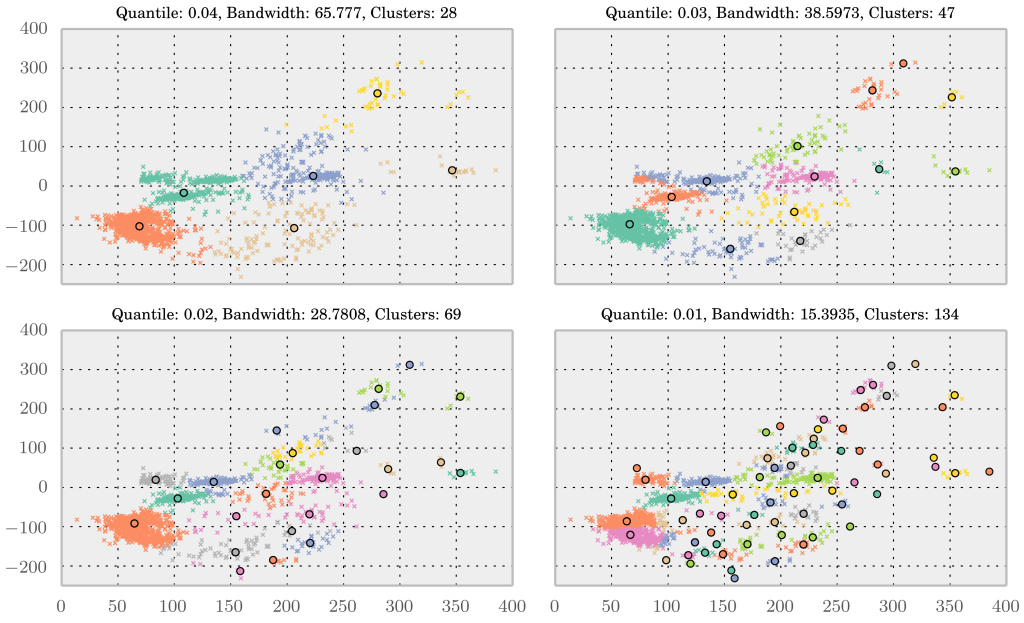



**Figure 4.8:** Visualising the outcome of mean shift clustering on samples less than 400W active power for varying Bandwidth values. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid.

## 4.6 DBSCAN

Density-based spatial clustering of applications with noise (DBSCAN) was proposed by Ester, Kriegel, Sander et al. in 1996 in [17].

Of particular interest to our problem domain is the DBSCAN definition of a cluster. In contrast to the techniques that we have already investigated, DBSCAN bases its notion of a cluster on density. Rather than basing a cluster on the notion of the relative distance of each point from the centroid of a set of local points, DBSCAN searches for regions in which the density of points is higher than the surrounding area [17]. This results in the ability of DBSCAN to discover arbitrarily shaped clusters.

Points in a cluster may be split into core-points and non-core (border) points. Core-points will be directly density reachable from each other. Border points may not be directly density-reachable from each other, but there must be a point within the core of the cluster from which both are density-reachable.

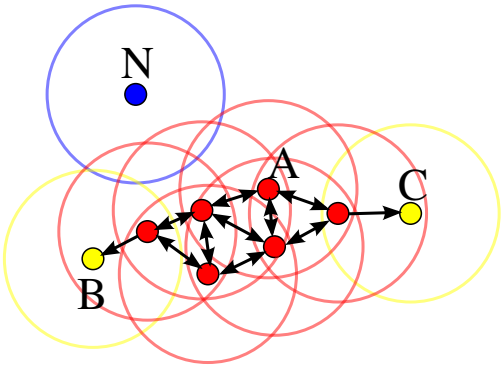



**Figure 4.9:** 'Points at A are core points. Points B and C are density-reachable from A and thus density-connected and belong to the same cluster. Point N is a noise point that is neither a core point nor density-reachable.(*minPts*=3 or *minPts*=4)'( [15])

Key to this approach are the parameters of the Eps-neighbourhood, $N_{Eps}$, which represents the distance from each point to consider its 'Neighbourhood' and the minimum number of points in a neighbourhood to consider a cluster, *minPts*.

A point, $p$, is directly density-reachable from another point $q$ if:

$$p \in N_{Eps}(q) \tag{4.4}$$

and

$$\mid N_{Eps}(q) \mid \geq minPts \tag{4.5}$$

DBSCAN also incorporates the concept of noise into its cluster model. Noise is defined as the set of points that do not belong to any cluster (where clusters have been discovered based on their density-reachability and density connectivity). A noise point will be neither density-reachable from a core-point nor indirectly density reachable via a chain of core-points. These concepts are illustrated in 4.9 reproduced from [15] under Wikimedia Commons license.

- **Advantages**
  - Implemented in `scikit-learn`, relatively straightforward to implement in our software.
  - Clusters can be of any shape.
  - Number of clusters does not have to be set in advance.
- **Disadvantages**
  - Two parameters to set: *minPts* and $N_{Eps}$.
  - Very sensitive to selection of parameters.
  - No dynamic execution (in `scikit-learn` implementation): clustering must be re-run if points are added.
  - Does not cluster well when clusters have very different densities (due to *minPts* and $N_{Eps}$ not generalising to all clusters).

### 4.6.1 DBSCAN Evaluation

We can again quantitatively assess the quality of the clustering obtained by DBSCAN by fitting our dataset with the algorithm whilst tuning the *minPts* and $N_{Eps}$ parameters.

Since DBSCAN clusters do not utilise the concept of a centroid in clustering, in order to visualise the clusters we proceed with the same colour-coding logic of cluster members as in the previous clustering examples, but add some post-processing steps.

1. We compute the mean of the core-samples and use this as a pseudo-centroid of the cluster when generating plots.

2. We colour-code the 'Noise' samples in black, as they do not belong to any clusters.

We choose an initial *minPts* figure of 8. Thus, our expectation is that a power-state must be detected by our edge detector at least 8 times before we can form a valid cluster. This is at the limit of the useful number of cycles for us to compute useful statistics about a power state. The selection of the *minPts* figure has a visible impact on the clustering ability of DBSCAN in our problem domain, since some appliances are much more active over a month than others. Consider again the many cycles of the refrigerator during a month, against the number of times a hoover is used.

Figure 4.10 shows that DBSCAN seems to perform well at clustering dense regions of points $>$ 500W active power even with high values for $N_{Eps}$. However, it exhibits similar drawbacks to the mean shift algorithm in that the region $<$ 500W active power fails to be discriminated into multiple clusters. Figure 4.11 shows in more detail the performance at clustering in this region. It can be seen that it is only when $N_{Eps}$ reaches 10 that clustering performance seems to improve in this region.

However, it is worth noting from Figure 4.10 in the lowermost-right plot that a cluster at approximately 1700W active power is classified as noise at this level, wheras it had previously been correctly identified by DBSCAN. This occurs because this cluster consists of very few samples that are quite spread out - thus the algorithm cannot find the desired *minPts* within the cluster when $N_{Eps}$ is 10. This is reflective of the expected performance of DBSCAN when cluster density is inhomogeneous across the dataset. Increasing the period of time over which data is collected and obtaining more samples should go some way to remedying this issue.

| *minPts* | $N_{Eps}$ | **MSC** | **# of Clusters** |
|---|---|---|---|
| 7 | 20 | 0.593001 | 14 |
| 8 | 20 | 0.588171 | 15 |
| 9 | 20 | 0.581679 | 16 |
| **8** | **11** | **0.501834** | **24** |
| 8 | 10 | 0.496187 | 23 |
| 7 | 9 | 0.492212 | 23 |
| 12 | 11 | 0.490673 | 17 |
| 9 | 10 | 0.488328 | 20 |
| 8 | 9 | 0.486896 | 18 |

**Table 4.3:** Optimising DBSCAN for Mean Silhouette Coefficient

In fact, if we investigate the ranges of: $5 \leq N_{Eps} \leq 20$ and $7 \leq minPts \leq 12$ we once again find that if we disregard the values when $N_{Eps}$ is 20 (as we have seen this clustering to fail in the lower power region) we find an 'island' of improved MSC that results in 24 clusters being discovered with $minPts = 8$ and $N_{Eps} = 11$ . Table 4.3 shows the highest results for MSC over the range described.

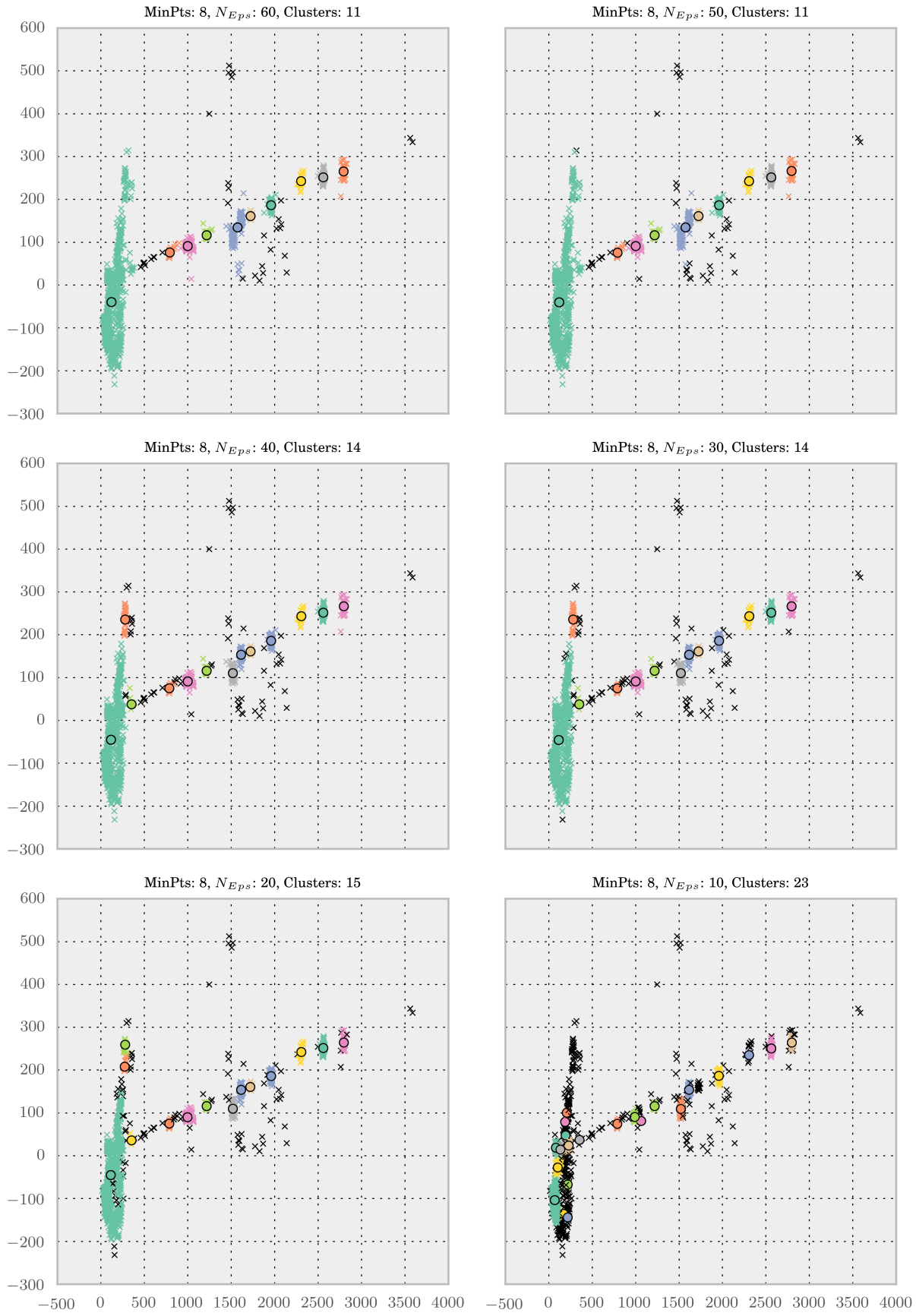



**Figure 4.10:** Visualising the outcome of DBSCAN for decreasing $N_{Eps}$ values. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid. Black ×'s are points classified as Noise by the algorithm.


**Figure 4.11:** Visualising the outcome of DBSCAN for decreasing $N_{Eps}$ values for $< 400$W active power. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid. Black ×'s are points classified as Noise by the algorithm.

## 4.7 The Hart Method - Incremental Multi-Variate Gaussian Mixture Clustering

In George Hart's 1985 and 1992 papers, [27] and [25], a method for the incremental/dynamic clustering of samples in the active/reactive signature-space is proposed.

The method models the signature space as containing a mixture of multi-variate Gaussian distributions, represented by ellipses within the signature space.

As samples are input into the algorithm, they are assessed for membership of any existing clusters by virtue of their inclusion within the ellipse of any existing cluster. If they do not fall within the ellipse of any existing cluster, a new cluster is created centred on the initial transition. The algorithm maintains two 'sub-clusters' within each cluster which provide an alternate model for the transitions within that cluster. The sub-clusters model the possibility that the master cluster is better explained as a mixture of two distributions than one. A statistical test is periodically triggered which evaluates the two models and splits a cluster if the two-distribution model is favoured. The same test is used to assess whether two master-clusters should be joined into one cluster if a sample is fed into the algorithm which falls within the ellipse of two master-clusters.

### 4.7.1 The HartCluster Object

The Cluster representation has been implemented as a Python Class which is illustrated in Figure C.2 in the Appendix C.

A cluster object created by the instantiation of the HartCluster class is shown in Figure 4.12. This object was constructed by randomly sampling from two multi-variate Gaussian distributions, intermixing the samples randomly, and then feeding them in to the object. Details of the distributions used are shown in the figure's caption.

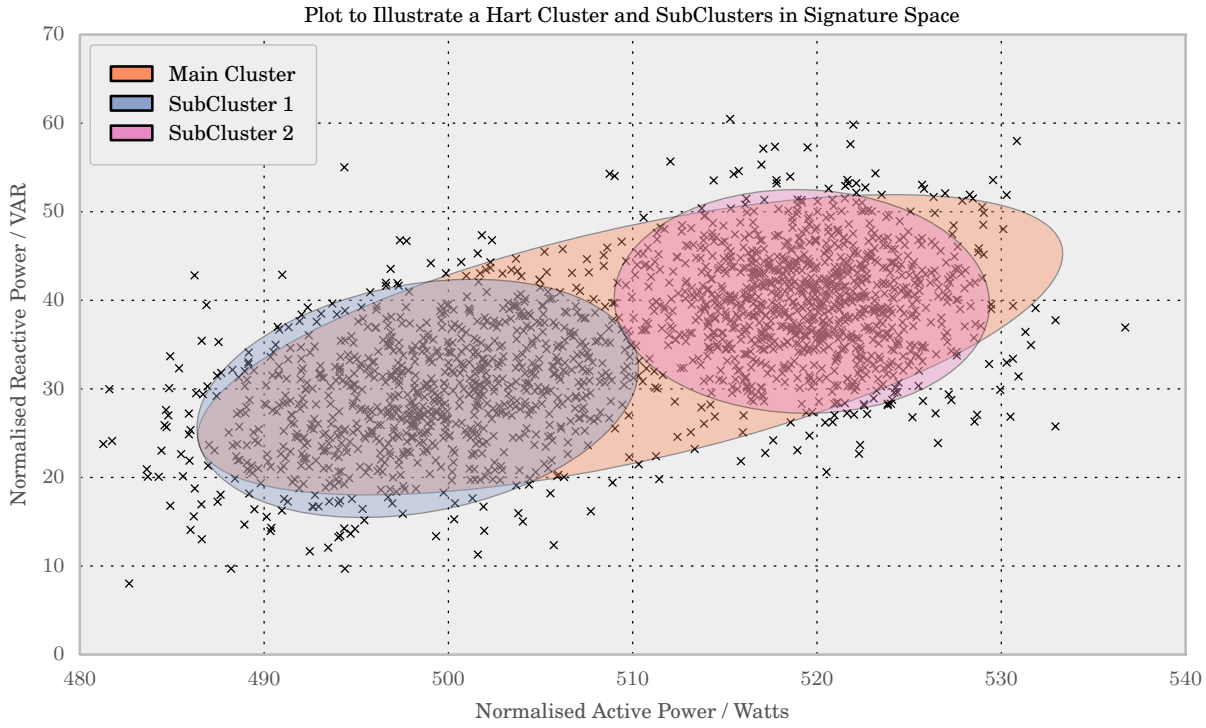



**Figure 4.12:** A HartCluster object and its primary and sub-clusters are visualised. The Ellipses are drawn at the 2-Sigma interval and the object was populated with 2000 samples, 1000 drawn from each of two multi-variate normal distributions and *presented to the Cluster in random order* parametrised with centroid: $(500, 30)$ and covariance-matrix: $\begin{bmatrix} 50 & 20 \\ 20 & 50 \end{bmatrix}$ and with centroid: $(520, 40)$ and covariance-matrix: $\begin{bmatrix} 20 & 0 \\ 0 & 40 \end{bmatrix}$.

It is clear from Figure 4.12 that whilst the main cluster is fitted well to the samples, the two

sub-clusters have correctly formed around the individual distributions. In fact the centroids of the sub-clusters have been computed as $(498.3, 28.9)$ and $(519.2, 39.9)$, illustrating that the sub-clusters are almost perfectly centred on the means of the two distributions that were sampled from. We may use the *splitJoinTest* that has been implemented as a class method to quantitatively assess whether a cluster is best explained by the two-distribution model of its sub-clusters, or the single model of the main-cluster.

Whilst Figure 4.12 illustrates a well discriminated case of two distributions being formed into one main cluster and two well fitted sub-clusters, it is worth noting that this is only the case because the two distributions were presented in a well mixed, random order of samples.

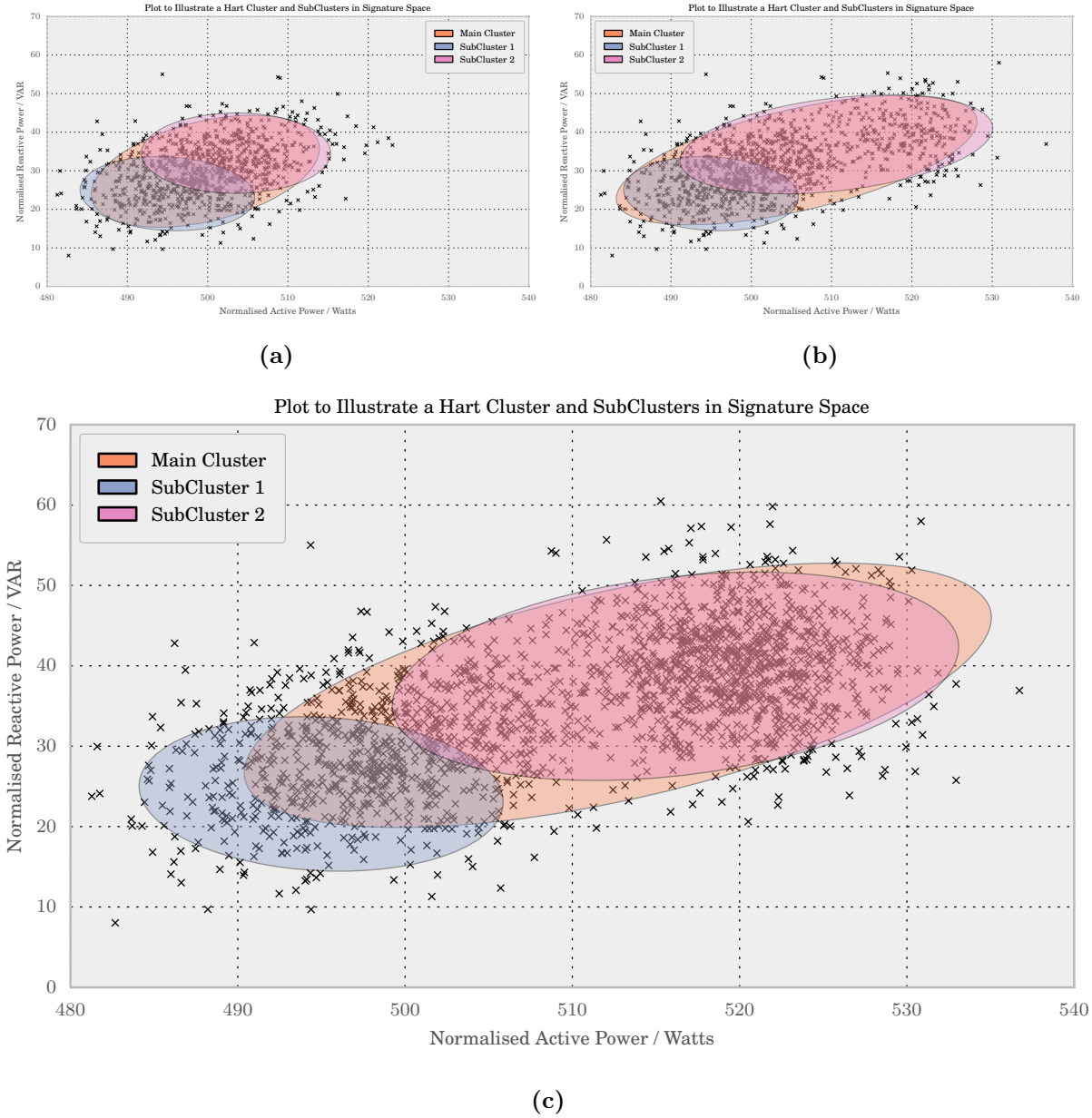



**Figure 4.13:** Visualising the development of Cluster and Sub-Clusters when two distributions are added to one cluster one after the other. **(a)** shows the state after the addition of one distribution. **(b)** shows the state after 250 samples from distribution 2 have been added. **(c)** shows the situation after 1000 points from both distributions have been added

As this algorithm is incremental, the clusters formed are dependent on the order of presentation of the samples. It is worth comparing Figure 4.12 against Figure 4.13c. Figure 4.13c was produced with the same samples (by fixing the random number generator's seed) but the individual distributions have been fed in to the Cluster object in order. Thus the Cluster has first formed around

the leftmost set of points, and maintained within itself two sub-clusters (when in reality there only exists one distribution). Then, once the second distribution is presented, the main cluster stretches to encompass the new distribution, and the sub-clusters begin to separate. One can clearly see in Figure 4.13c that this results in Sub-Cluster 1 being more to the lower left of the figure, as it was originally the 'left hemisphere' of the 1st distribution fed in, whilst the second Sub-Cluster encompasses both distributions, though it has shifted significantly toward the second distribution.

Sub-figures 4.13a and 4.13b show the intermediate cluster states. Sub-figure 4.13a shows the Cluster object after 1000 samples from the first distribution are added. Sub-figure 4.13b shows the intermediate state where the first 250 entries of the second distribution are added. Figure 4.13c is the final state after both distributions have been added.

The discussion above illustrates that when presented with randomly ordered data from two distributions a HartCluster object is able to track both the independent models of each distribution as a Gaussian distribution, and a combined model of both distributions.

We have also seen that the HartCluster class as implemented is able to adapt when new samples drawn from a different distribution to that which it was originally populated with are presented to it. This gives us the useful property of gradually moving to a new fit to the data upon receipt of new samples. As our situation involves presenting samples for clustering that are received in strictly time-ordered fashion, the implementation should allow for slight changes in the power-state distribution of an appliance over time to be adapted to. This is observed by Hart in [25]. In order to improve the ability to do this, a 'finite memory filter' is implemented within this class. The filter gives greater weight to the latest input samples in calculating the covariance-matrix and mean for the distribution. The 'bias' of the filter's weights and the number, $N$ of samples to consider 'recent' can be set at runtime.

As noted by Hart in [25], parametrising a distribution with very few points sampled from within it can result in a misclassification. Thus, the HartCluster class is instantiated with a *minPts* parameter, which determines the minimum number of points that must belong to the cluster before it will attempt to compute a covariance matrix for the distribution. Until this number of points is reached, membership of the cluster is tested by determining if a sample falls within a fixed radius from the mean of the samples already in the cluster. This radius is set to a value of 20 (which can be imagined as an initial circular 20 Watt or VAR region around a point)

### 4.7.2 The Split-Join Test

The statistical test which is used to decide whether to join two clusters or split one cluster into two is described in Hart 1985 (p. D1-D2) [25] and reproduced here. The argument $N_i$ is the number of observations in the $i$th cluster, the argument $M_i$ is the mean of the observations of the $i$th cluster (i.e, the centroid of the ellipse) and $S_i$ is the covariance matrix of $i$th cluster. Thus, the combined distribution of the two input clusters is subscripted 3.

The likelihood ratio, $L$ indicates the relative likelihood of the one cluster hypothesis over the two cluster hypothesis.

$$N_3 = N_1 + N_2 \tag{4.6}$$

$$M_3 = \frac{N_1 M_1 + N_2 M_2}{N_3} \tag{4.7}$$

$$S_3 = \frac{N_1 S_1 + N_2 S_2}{N_3} + \frac{N_1 N_2}{{N_3}^2}(M_1 - M_2)(M_1 - M_2)^T \tag{4.8}$$

$$L = \frac{N_1}{N_3}\ln(\det S_1) + \frac{N_2}{N_3}\ln(\det S_2) - ln(\det S_3) \tag{4.9}$$

In [25], Hart found that the thresholds for action on his data and software implementation were $L = -2.5$ and $L = -3.5$. That is, two clusters are joined if $L \geq 2.5$ and two sub-clusters are merged if $L \leq -3.5$. This was determined empirically by observing the results of applying the test to clusters known to belong together or separate.

The gap between the values of L enforces a hysteresis in the algorithm, and prevents clusters being repeatedly joined and split.

Testing has been undertaken by the author to determine the appropriate values to use in his implementation. Figure 4.14 illustrates the computed values of $L$, the Likelihood ratio for two distributions as one approaches and is eventually subsumed into the other. A combination of visual and computational tests were undertaken to 'tune' these parameters so as to result in the best clustering performance. As a result, the thresholds for action using the author's implementation of this clustering method have been set at $L = -2.0$ and $L = -3.0$.

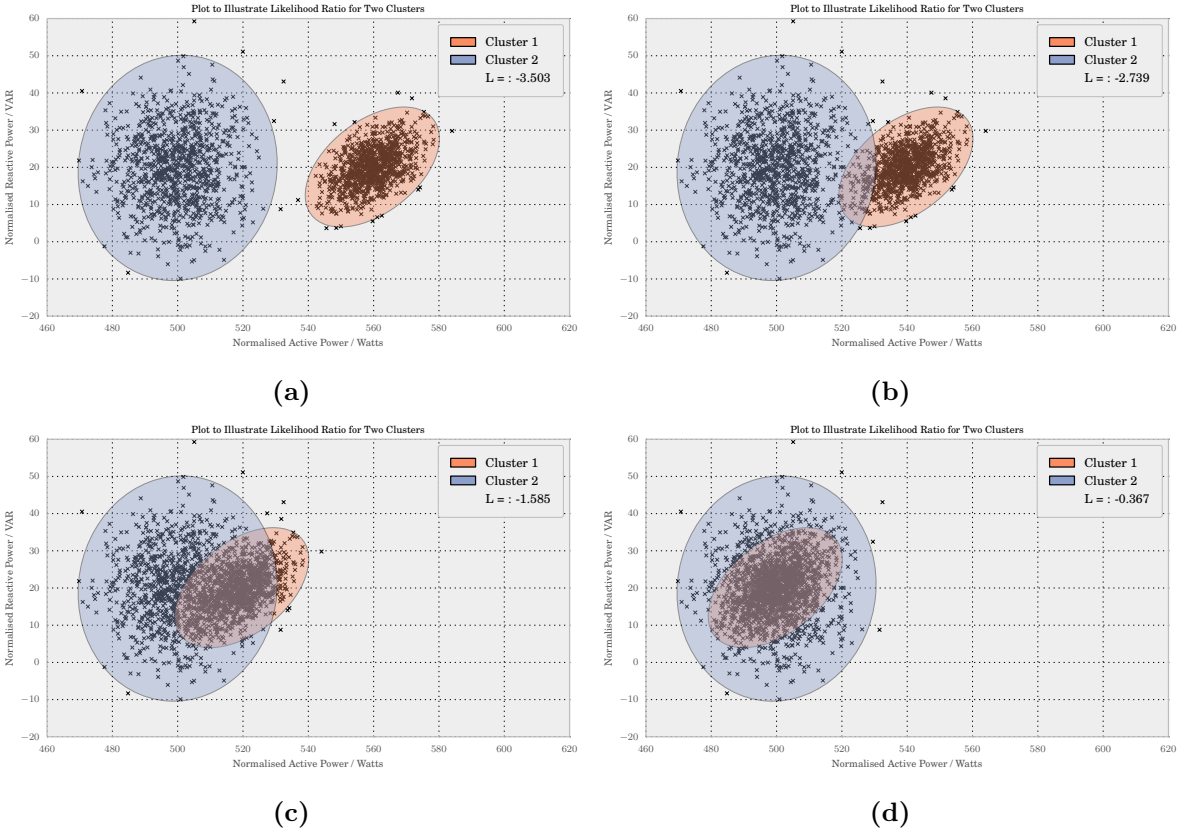



**Figure 4.14:** Illustrating the result of the split-join test for two clusters, initially representing two clearly distinct distributions and observing the result of subsuming one distribution into the other on the Likelihood ratio, $L$.

### 4.7.3 The Clustering Algorithm

Algorithm 4 shows the steps involved in the clustering. The output of the algorithm is a list of HartCluster objects. In addition to the filter-length and the minimum-points to wait for before computing the distribution covariance matrix, the algorithm requires us to set the 'range' at which to consider a point within a cluster. This is achieved by selecting the number of standard-deviations *nStd* at which to draw the ellipse from the centroid of the distribution. In [25] an analysis is performed which sets the minimal value for our number of dimensions of $nStd = 2$. Experimentation with the algorithm has resulted in a value of $nStd = 3$ providing best results when assessed with a Mean Silhouette Coefficient.

**Algorithm 4** Hart Clustering Algorithm

```
1:  state_list                                   ▷ List of power states generated by PairBuffer class
2:  f_length ← 25                                                    ▷ Length of finite memory filter
3:  minPts ← 10          ▷ Number of points to require before parametrising distribution
4:  clusters ← [HartCluster(state_list[0], f_length, minPts)]
5:  count ← 1
6:  joinLimit ← −1.4                                                 ▷ Test limit for joining clusters
7:  splitLimit ← −2.5                                                ▷ Test limit for splitting clusters
8:  for state in state_list[1 :] do
9:      if len(clusters) < 1 then
10:         if clusters[0].stateInCluster(state) then
11:             clusters[0].addState(state)
12:         else
13:             clusters.append(HartCluster(state, f_length, minPts))
14:         end if
15:     else
16:         close1, close2 ← getClosestClusters(state, clusters)
17:         if close1.stateInCluster(state) then
18:             if close2.stateInCluster(state) then      ▷ If state in 2 clusters, test joining them
19:                 L ← splitJoinTest(close1, close2)
20:                 if L > joinLimit then
21:                     newCluster ← joinClusters(close1, close2)
22:                     clusters.remove([close1, close2])          ▷ Remove old clusters from list
23:                     clusters.append(newCluster.addState(state))
24:                 else
25:                     close1.addState(state)                     ▷ Add state to nearest cluster
26:                 end if
27:             else
28:                 close1.addState(state)
29:             end if
30:         else                          ▷ State did not belong in existing cluster, add a new one
31:             clusters.append(HartCluster(state, f_length, minPts))
32:         end if
33:     end if
34:     count ← count + 1
35:     if ¬(count mod 25) then      ▷ Every 25 states added, test if any clusters should be split
36:         clusters = testSplits(clusters)
37:     end if
38: end for
    return clusters
```

The *testSplits*() function that is called on line 36 of the algorithm checks each HartCluster in the list according to the Split-Join test described above and if supported by the Likelihood ratio,

splits the Cluster into two by promoting the sub-clusters to be 'full' clusters, constructing new sub-clusters within them, and proceeding to allocate transitions to the new sub-clusters. This process is illustrated in the code extract in Figure 4.15 on page 61.

```
def splitCluster(cls, cluster):
    # get two cluster references, unset as subclusters
    newClusterOne = cluster.subC1.setSubCluster(False)
    newClusterTwo = cluster.subC2.setSubCluster(False)

    for c in [newClusterOne, newClusterTwo]:
      # Instantiate SubClusters and insert
      #  transitions into appropriate one.
        c.subC1 = HartCluster(transition = c.transitions[0],
                              fLength = c._fLength,
                              SubCluster=True,
                              minEPoints=c._minEPoints
                             )
        if len(c.transitions > 1):
            c.subC2 = HartCluster(transition = c.transitions[1],
                                  fLength = c._fLength,
                                  SubCluster=True,
                                  minEPoints=c._minEPoints
                                 )
            for t in c.transitions[2:]:
                # Add transition to nearest subcluster
                nearest = getClosestClusters(t,[c.subC1, c.subC2])[0]
                nearest.addTransition(t)

    return newClusterOne, newClusterTwo
```

**Figure 4.15:** Python code extract illustrating the process of splitting a Cluster and creating two new Clusters. The newly created sub-clusters inherit the *f_length* and *minPts* from the split Cluster.

The process for joining clusters is rather simpler: if the Split-Join test determines that clusters should be merged, the two clusters become sub-clusters of a new master-cluster, which is instantiated with the new parameters for its centroid, $M$ and covariance matrix, $S$ that were determined by the Split-Join test.

- **Advantages**
  - Designed originally for this problem domain.
  - Clusters modelled as multi-variate Gaussian distributions. Suited to our device model.
  - Number of clusters does not have to be set in advance.
  - Can be run in a dynamic fashion.
  - Caters for clusters of varying density.
- **Disadvantages**
  - Multiple parameters to set: *minPts*, *nStd*, *f_length*.

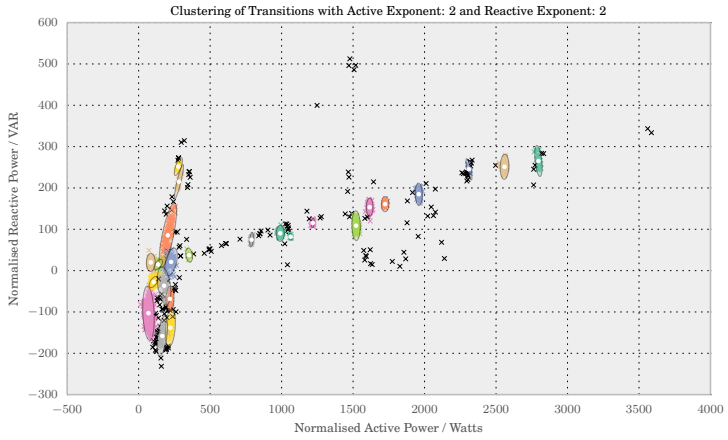
### 4.7.4 Hart Clustering Method Evaluation

The clustering algorithm as implemented returns labelling data for each of the transitions that have been input, and thus is able to be evaluated using the `scikit-learn` Silhouette Score functions.

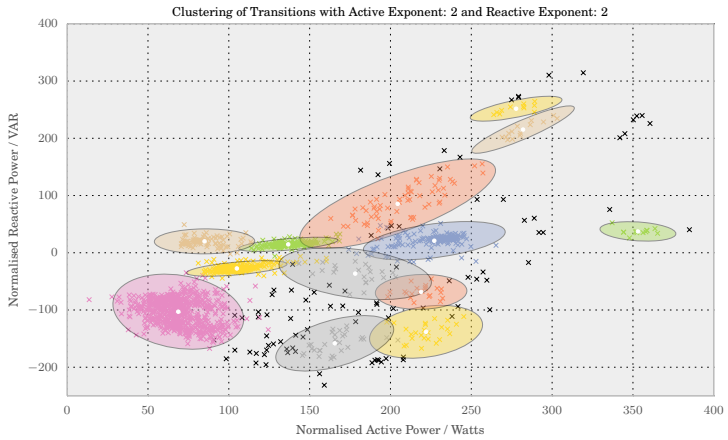
| *minPts* | **Filter Length** | **MSC** | **Clusters** |
|---|---|---|---|
| 10 | 30 | 0.600091 | 26 |
| 10 | 32 | 0.59956 | 26 |
| 10 | 34 | 0.59956 | 26 |
| 10 | 36 | 0.59956 | 26 |
| 10 | 38 | 0.59956 | 26 |
| 10 | 40 | 0.59956 | 26 |
| 11 | 30 | 0.595956 | 24 |
| 11 | 32 | 0.595426 | 24 |
| 11 | 34 | 0.595426 | 24 |

**Table 4.4:** Highest 10 Mean Silhouette Coefficients achieved when clustering with Hart's 1985 method scored in the ranges $22 \leq f\_length \leq 40$ and $8 \leq minPts \leq 12$

The algorithm performs extremely well in clustering the same data that was presented to the algorithms discussed above. We may again compute the MSC for a variety of values of input parameters. We resolve to use a $nStd$ of 3 when seeking to compute whether a sample falls within a cluster. If we consider the following ranges for the two other tune-able parameters, $22 \leq f\_length \leq 40$ and $8 \leq minPts \leq 12$, we find that the greatest MSC is obtained for $minPts = 10$ and $f\_length = 30$. Interestingly, the difference in silhouette scores for values in this range is minimal. The algorithm seems to perform well and is relatively insensitive to changes in these inputs. As would be expected, the number of 'valid' clusters produced is inversely correlated to the *minPts* figures. The highest 10 values for the MSC achieved in the ranges of parameters specified above are shown in Table 4.4. In fact, the minimum MSC achieved over the whole range was 0.567. The reason for the relative insensitivity of the algorithm to the filter length may be due to many of clusters having a number of samples similar to $f\_length$ within them, and thus all samples are treated with equal weight.




**(a)**




**(b)**

**Figure 4.16:** Plots illustrating the performance of the Hart clustering method on our dataset. Black ×'s are points belonging to clusters with fewer than 10 points. Clusters are shown as colour-coded ellipses at the 3rd standard deviation interval. Cluster centroids are marked with white circles with a black edge.

## 4.8 Summary

Four clustering algorithms have been assessed for their suitability and performance in extracting meaningful clusters from our normalised, averaged power-states.

We have made use of three `scikit-learn` library implementations of algorithms (which have been extended to generate appropriate output to allow for disaggregation of power), and a library implementation of cluster-scoring metrics.

The author has implemented the incremental Gaussian mixture clustering method described in [25] in Python, with the objective of releasing the source-code to the community, and assisting in its implementation in NILMTK. The implementation allows for the clustering to be assessed using `scikit-learn` cluster scoring metrics.

We have seen that k-means clustering suffers the drawback that we must specify the number of clusters, $K$, in advance, and that the act of establishing an optimum value for $K$ is complicated by the fact that the algorithm tends to produce clusters of even size, and models clusters as spherical. We must make an assumption that low values of $K$ are likely to result in poor identification of clustered-power states and thus search for good clustering results (as established using a mean silhouette coefficient for values of $K$ above 20). Searching for an optimum value for $K$ requires running the algorithm multiple times, which can take a considerable amount of time with a large dataset.

Mean shift clustering was seen to be very sensitive to selection of the Bandwidth parameter. We established that optimising the Bandwidth parameter required us to investigate a large range of Bandwidth values, and that, similarly to k-means clustering, the algorithm struggled with our differently sized clusters and amount of noise. We also observed that the selection of kernel with which to assess clusters had an impact on clustering performance. We noted that as Mean shift requires many nearest-neighbour searches it does not scale well to large N.

We noted that DBSCAN was very sensitive to its parameters but that its concept of noise was beneficial in enabling it to distinguish dense clusters in a noisy field in the lower power regions of our data. It is worth remarking that the classification of less dense regions as noise can have a draw-back in disaggregation as it excludes many observed power-states from clustering. We also found that in tuning DBSCAN to identify clusters in our lower power range, we increased the number of points treated as noise at higher power ranges, where they were previously clustered.

We have seen that the Hart method of clustering was designed for our data, can be run dynamically, models well our power-states in the signature space, and performs well in identifying clusters of varying densities and orientations in our signature-space. We note that there are multiple parameters to be selected, but that the algorithm appears relatively consistent in clustering as these are varied. It is also fast to optimise as the clustering requires one pass over the data. We can also allow small clusters to be treated as noise, but allow for these to be included if necessary when mapping clusters of power states to devices.

We shall compare the disaggregation performance of each of the algorithms in Chapter 7.

# Chapter 5

# Matching Clustered Data to Appliances

Once our power-states have been clustered into groups containing those states with alike reactive and active power, we can begin the process of identifying the devices within the home to which those power-states belong.

First, we post-process the clustered samples to produce a `pandas` DataFrame object containing time-ordered data on the power-states belonging to each cluster and the paired transitions that comprise the power-state. We also compute the duration of the state, and the energy consumed (in kWh) during the state. The DataFrame contains all samples that fell within the cluster, but also contains those samples to which the cluster centroid is 'nearest' (as computed by Euclidean distance) but that were 'orphaned' as a result of initially belonging to a cluster with too few samples to consider it 'valid' or were classified as noise by the clustering algorithm. We store a Boolean to indicate whether the sample is truly within the cluster or is associated with it by virtue of its proximity (as in the orphaned, or noise case).

The `pandas` DataFrame is a two-dimensional size-mutable, potentially heterogeneous tabular data structure with labelled axes (rows and columns). Arithmetic operations can be computed on both row and column labels [42]. The DataFrame is a fantastically useful class for our application as it simplifies many of the operations that are required to align and match our time-series data. It also allows for the output of its contents to a LaTeX table format, as is shown in Table 5.1.

| | T1 Time | T1 Active | T1 Reactive | T2 Time | T2 Active | T2 Reactive | Average Active Power | Average Reactive Power | State Duration | Energy Consumed |
|---|---|---|---|---|---|---|---|---|---|---|
| 359 | 2014-07-16 07:43:25.400000+01:00 | 1063.143 | 82.00462 | 2014-07-16 07:46:31.200000+01:00 | -1061.548 | -81.11106 | 1062.345351 | 81.557840 | 185.8 | 0.054829 |
| 1377 | 2014-07-25 09:54:22.900000+01:00 | 1058.889 | 84.69289 | 2014-07-25 09:56:00.800000+01:00 | -1056.611 | -93.21878 | 1057.749899 | 88.955832 | 97.9 | 0.028765 |
| 2153 | 2014-08-01 07:45:41+01:00 | 1066.415 | 79.74586 | 2014-08-01 07:48:36.900000+01:00 | -1065.419 | -76.64255 | 1065.916780 | 78.194207 | 175.9 | 0.052082 |
| 2590 | 2014-08-06 06:30:12.600000+01:00 | 1057.788 | 80.70635 | 2014-08-06 06:30:49.600000+01:00 | -1071.787 | -77.40265 | 1064.787300 | 79.054500 | 37.0 | 0.010944 |
| 2591 | 2014-08-06 06:34:07.400000+01:00 | 1051.305 | 75.96264 | 2014-08-06 06:36:13.300000+01:00 | -1063.169 | -83.75705 | 1057.236956 | 79.859849 | 125.9 | 0.036974 |
| 2592 | 2014-08-06 06:36:23.300000+01:00 | 1066.379 | 81.12981 | 2014-08-06 06:36:41.300000+01:00 | -1065.515 | -80.76362 | 1065.947182 | 80.946714 | 18.0 | 0.005330 |
| 2593 | 2014-08-06 06:36:47.300000+01:00 | 1065.75 | 84.68306 | 2014-08-06 06:37:04.300000+01:00 | -1065.435 | -79.80111 | 1065.592810 | 82.242089 | 17.0 | 0.005032 |
| 2594 | 2014-08-06 06:37:10.300000+01:00 | 1063.717 | 83.48339 | 2014-08-06 06:37:45.300000+01:00 | -1064.677 | -84.1401 | 1064.197279 | 83.811745 | 35.0 | 0.010346 |
| 2595 | 2014-08-06 06:37:56.300000+01:00 | 1065.169 | 79.61327 | 2014-08-06 06:38:18.300000+01:00 | -1065.83 | -82.02863 | 1065.499828 | 80.820954 | 22.0 | 0.006511 |
| 2596 | 2014-08-06 06:38:30.300000+01:00 | 1067.517 | 89.69651 | 2014-08-06 06:38:36.300000+01:00 | -1063.495 | -85.53391 | 1065.506115 | 87.615211 | 6.0 | 0.001776 |
| 2597 | 2014-08-06 06:38:45.300000+01:00 | 1060.967 | 78.63461 | 2014-08-06 06:38:55.200000+01:00 | -1064.946 | -82.77407 | 1062.956331 | 80.704342 | 9.9 | 0.002923 |
| 2598 | 2014-08-06 06:39:09.200000+01:00 | 1067.586 | 80.96598 | 2014-08-06 06:40:52.200000+01:00 | -1060.679 | -80.06316 | 1064.132452 | 80.514570 | 103.0 | 0.030446 |

**Table 5.1:** A pandas DataFrame representing those samples included in a cluster at around 1050W active power.

The data stored for each cluster allows for a further piece of post-processing: using the time-stamps of each ON/OFF transition we can re-construct a time-series representation of the power states of a cluster at an interval of our choosing. This will allows us to compare the cluster's power

consumption against 'ground truth' measurements from individual meters.

As was discussed in 2.4, as well as collecting data at 1Hz frequency on the primary supply to the household, many (though not all) devices have individual appliance monitors transmitting data about their individual consumption to our data collection PC.

## 5.1 Using Appliance Ground Truth Measurements

### 5.1.1 Preparing Appliance Level Data

Appliance level data is received at approximately 6-second intervals (see Section 2.4 for more detail) and the power data that is transmitted is dependent on the appliance level monitor. Current transformer meters (as used in the author's home on the kitchen equipment where devices are not connected via a socket) record *apparent* power in VA. Current transformer type meters only sample current, and they must use a hard-coded value for voltage in determining instantaneous power consumption. As we have seen that voltage can vary by quite a wide margin, the values reported for power from the CT meters my vary of a range of +20% to −12% [36]. Individual appliance monitor transmitter plugs (as used on any device connected to the mains via a plug-socket) record *active* power measurements, in Watts.

All appliance level data is loaded from .csv files into `pandas` DataFrames. The time-stamps are converted from UNIX time-stamp format into time-zone localised and aware `pandas` DatetimeIndex objects. This ensures that any measurements that pass over BST/GMT dates are correctly interpreted when manipulated.

Data alignment and analysis requires pre-processing of data, since data packets from the wireless transmitters are occasionally lost, we must pre-process appliance level data to fill any short gaps in sample intervals. As meters may lose power, or be turned off, a long gap between data samples is treated as if the appliance was off.

| | |
|---|---|
| 24_inch_lcd | nespresso_pixie |
| 24_inch_lcd_bedroom | network_attached_storage |
| atom_pc | oven * |
| core2_server | primary_tv |
| dishwasher * | sky_hd_box |
| electric_hob * | steam_iron |
| fridge_freezer * | stereo_speakers_bedroom |
| hairdryer | toaster |
| home_theatre_amp | treadmill |
| i7_desktop | vacuum_cleaner |
| kettle | washer_dryer * |
| microwave * | |

**Table 5.2:** Appliances for which data was collected using individual monitors. An asterisk indicates a CT type monitor was used.

In order to pad any regions in our time series where samples are missing for long periods (defined as four multiples of the sample rate, i.e. 24 seconds in this case) we make use of a function from **nilmtk.preprocessing.electricity.single**, **insert_zeros()**. Once zero values have been inserted around the regions where data was not recorded, we can then 'resample' the measurements in the DataFrame to fit any time interval we desire, using in-built `pandas` functions. We resample to 1Hz measurements by forward-filling the last known value at any time into the missing time intervals.

Samples are also screened according to a high and low 'threshold'. Since an appliance may draw a 'vampire-power' of a few Watts, even when in an ostensibly OFF state, we wish to screen our Appliance level data so as to treat a device as OFF when it is not in a truly ON power-state. Due to the relative inaccuracies of the CT style meters at at powers below 200W (see [36] p.11) and the tendency of kitchen appliances to have relatively high Wattage power-states the OFF thresholds for some of these devices are relatively high. We use the 'high' threshold to screen any erroneous 'blips' of unreasonably high power consumption that are on occasion sent by the monitors.

```
1  def prepareAppliance(appliance, threshold):
2
3      # Use the function from nilmtk to insert zeroes around periods where we've not
       got data
4      appliance = nilmtk.preprocessing.electricity.single.insert_zeros(appliance)
5
6      # Forward fill
7      appliance = (appliance[(appliance >= threshold[0]) &
8                  (appliance <= threshold[1])]).icol(0).dropna()
9
10     # Round the index to the nearest second
11     appliance = _roundIndexToSeconds(appliance)
12
13     # Resample and localise new appliance frame
14     applianceResampled = appliance.resample('S', fill_method='ffill')
15                                   .tz_localize('UTC', infer_dst=True)
16                                   .tz_convert('Europe/London')
17
18     return applianceResampled
```

**Figure 5.1:** Simplified Python code extract illustrating the process of pre-processing appliance level power data.

The final output of our appliance processing results in appliance-level data, converted to 1 second intervals of samples, screened of vampire power and infeasibly high power values. The time stamps are to the nearest second, rather than at whatever millisecond the value happened to be sampled. This will allow alignment with post-processed data from our clusters. A simplified extract of the code used to achieve this processing is shown in Figure 5.1.

### 5.1.2 Preparing Clustered Data

Each cluster discovered by the clustering algorithm is then processed to produce a DataFrame equivalent in content to the output of our individual appliance pre-processing procedure.

First, the ON and OFF times and average active power of each power-state in the cluster are extracted from the cluster DataFrame (similar to that shown in Table 5.1) and used to form a time ordered series of the start and end times of the power. The start of one such series formed by this process is shown in Table 5.3. This cluster contains the power-states to the fridge-freezer.

| Time | average active power |
|---|---|
| 2014-07-12 23:35:46 | 97.31 |
| 2014-07-12 23:57:22 | 0.00 |
| 2014-07-13 00:37:20 | 98.80 |
| 2014-07-13 00:59:17 | 0.00 |
| 2014-07-13 01:39:43 | 104.83 |
| 2014-07-13 02:02:12 | 0.00 |
| 2014-07-13 02:43:20 | 104.96 |
| 2014-07-13 03:09:05 | 0.00 |
| 2014-07-13 03:51:15 | 100.95 |
| 2014-07-13 04:12:04 | 0.00 |
| ... | ... |

**Table 5.3:** ON-OFF sequence for a Cluster.

The same code as used for appliance prepartion (on lines 11 and 14 in Figure 5.1) can then be deployed on this series to 'reconstruct' a second-by-second series of power-states for all of the samples in the Cluster.

This process is run on every cluster produced by the clustering algorithm and results in reconstructed time-series for the power-states within the cluster. We may then proceed to compare the power-activity of the cluster with that of the appliances in order to map a cluster onto the most likely appliance to have produced the states recorded within it.

Figure 5.2 shows the fridge-freezer time-series from an individual appliance monitor and a reconstructed cluster-time series generated by the edge detection, pairing procedure, and clustering. It is possible to see that the majority of power states throughout a 24 hour period are coincident.

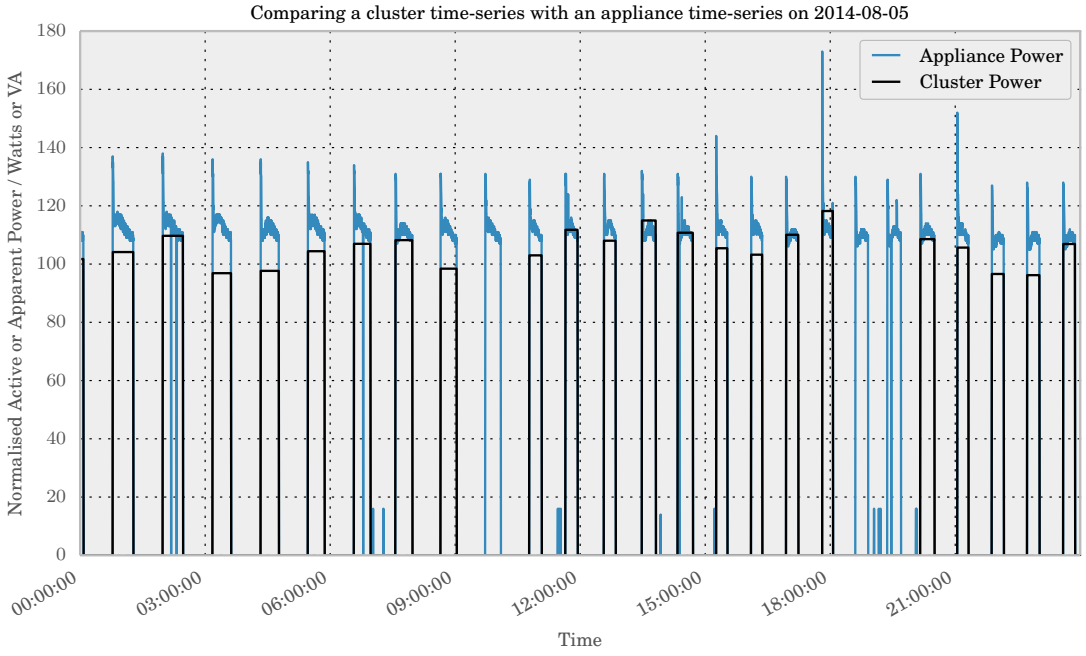



**Figure 5.2:** The apparent power recorded by the fridge-freezer CT monitor is shown against the power-states captured within a cluster extracted from the active/reactive signature space.

### 5.1.3 Assigning Power to Appliances

The initial effort at matching our cluster time-series data to ground-truth appliance level time-series data used a least-square difference test. The time-series data containing the reconstructed power-levels from each cluster was 'joined' to the data from each appliance time-series in turn, and the difference between the instantaneous active or apparent power subtracted (depending on the monitor used for the appliance). This difference at each sample was squared, and then the summed, to form a chi-squared test. The appliance for which this figure was minimised was selected as the candidate which produced the power-states recorded in the cluster.

The manner in which the time-series of the clusters and appliances are aligned is important. Since we expect that appliances will in general have multiple power-states, but that each cluster will represent only a single power-state, we undertake a *LEFT* join of the cluster and appliance time series only at those times when the cluster is in an ON power-state. This ensures that we only compare those times for the appliance with those when the cluster was in an ON state.

If a *FULL* join is made between the appliance and cluster time series, we will be comparing *EVERY* power-state of the appliance against the cluster. This is, in the general case of a multi-state appliance, a rather poor comparison to make. The square difference will (hopefully) be small for those times when the appliance is actually in the state which has been captured by the cluster, but it will be very large for those periods where the appliance is in a different state, and the cluster is essentially seen as 'OFF' as it only represents one state.

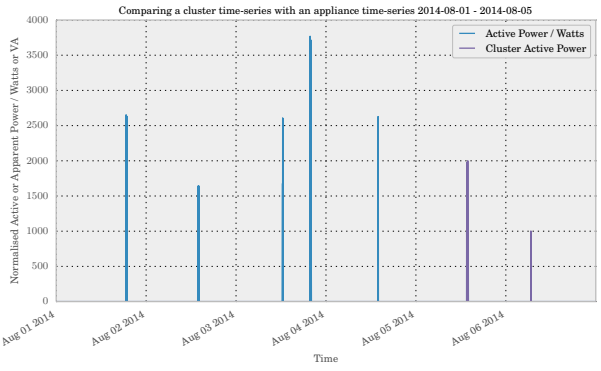



**Figure 5.3:** Comparing the full ground-truth record for the Electric Hob with one cluster

This is illustrated in Figure 5.3 which compares the time-series data from the electric hob. This is a typical multi state device, it has 5 burners, each with a different power-consumption. Each burner should be represented by one cluster. If we compare one cluster (which

represents a burner element) with the *entire* collection of appliance power-states the difference between the series will be great, as the cluster only captures the 6th and 7th ON states of the burner in the figure.

### 5.1.4 Quantifying the Energy Assignment

In order to assess our success at assigning power to devices, metrics from NILMTK [4] are used to compare the energy assigned to each appliance with the ground truth measurements for each appliance. We make use of the following metrics in the discussion that follows:

1. **Fraction of energy assigned correctly** The overlap between the fraction of energy assigned to each appliance and the actual fraction of energy consumed by each appliance over the dataset.

$$fraction = \sum_{n} min\left(\frac{\sum_n y}{\sum_{n,t} y}, \frac{\sum_n \hat{y}}{\sum_{n,t} \hat{y}}\right) \tag{5.1}$$

2. **Error in total energy assigned** The difference between the total assigned energy and the actual energy consumed by appliance $n$ over the entire data set.

$$error^{(n)} = \left|\sum_t y_t^{(n)} - \sum_t \hat{y}_t^{(n)}\right| \tag{5.2}$$

Using data collected in the author's home over the period 2014-07-13 - 2014-08-08 we find the following results after edge detection, pairing, clustering using Hart's method and using the chi-square method to assign power to appliances:

| | |
|---|---|
| Total Energy Accounted for within Clusters / kWh | 90 |
| Total Energy Sub-metered / kWh | 261 |
| Total Aggregate Energy Use / kWh | 426 |
| Fraction of Power Detected | 0.211 |
| Fraction of Energy Assigned Correctly | 0.471 |

**Table 5.4:** Results for energy attribution using Hart Method and Least-Squares assignment method.

On initial inspection, these results do not seem very successful. However, there are a number of factors worth turning our consideration to which give us some hope that the Edge Detection approach to disaggregation is not an abject failure!

Inspecting Figure 5.4 we can begin to understand more intuitively where energy has been assigned, and where our edge detection and clustering approach is successful, and where it has failed.

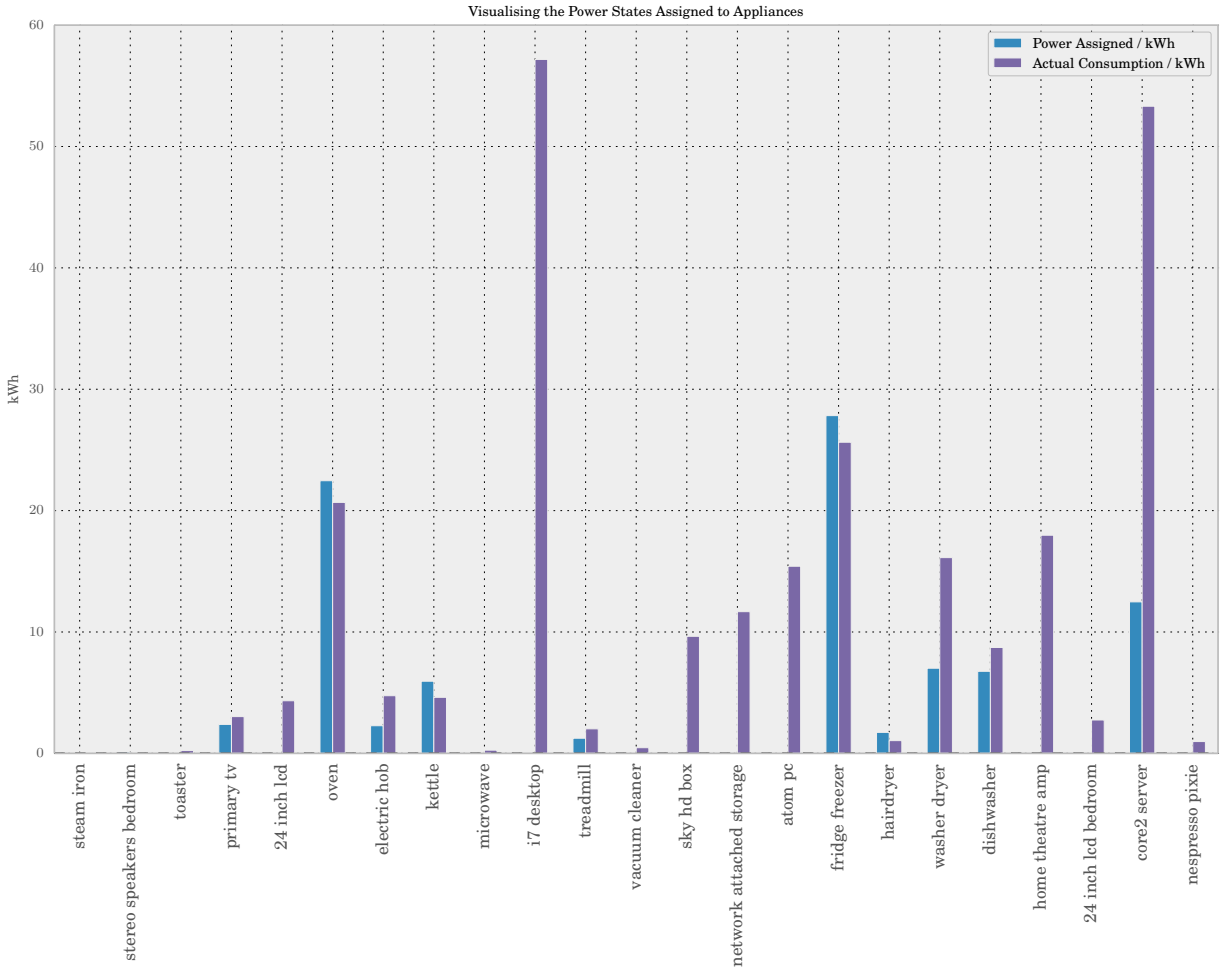



**Figure 5.4:** Energy Assigned to Devices using Hart Clustering Method and Least-Squares Matching.

- **Large power consumption of PCs in the household**

  First, it is immediately evident that of the sub-metered appliances, two of the PCs in the household consumed about one quarter of the total energy used over the period! This is a problem for our disaggregation approach: we are attempting to detect Edges in the power-signal being monitored at the home. PCs are continuously-variable devices. Aside from the initial ON and eventual OFF transition, as the graphics card or processor is more taxed, the power-consumption rises. Over a typical use cycle, one can expect gradual undulations in the CPU and GPU use, and thus gradual changes in the power-consumption of the device. These, if they are detected by our edge-detector, will likely be thrown out during the pairing-process.

- **Assignment of energy to continuously variable devices which are predominantly in an ON state**

  In spite of the fact that we do not expect to detect the PCs, the core2 server is assigned over 10kWh of energy. The appliance is always on, and consumes between 70W and 400W of power (though the vast majority of its time is spent in the lower end of this range). As we will see, some appliances produce nearly identical power-states. This causes clusters to contain power-states belonging to multiple appliances. Such clusters can then not be matched to individual appliances well when using the least-squares approach. In this situation, the characteristic of the core2 server of being always on causes it to be assigned clusters erroneously - because when no particular device is well correlated to a cluster containing multiple states, the server's power consumption reduces the least-squares distance from the cluster, and it is assigned the energy.

- **Lack of comprehensive ground truth measurements**

It is possible to see from Table 5.4 that the total sub-metered power leaves a large proportion of the energy unaccounted for. A large portion of this would most likely be attributed to the lighting circuit. In fact, it is likely that the clusters at around 80W and 120W active power are in reality lights. 40W Halogen bulbs are in use throughout the home in 2 and 3 bulb light fittings. These clusters are most likely to be assigned to the Fridge-freezer or core2 server, and explain to some degree the unexpected power assigned to these devices.

- **Relative success at assigning power to Fridge-freezer, Dishwasher, and Oven**

  Figure 5.4 shows that the power assigned to these three devices looks quite accurate. As these are three rather dissimilar devices with differing use and power-state profiles, this provides evidence of the generalisability of the algorithm across at least a few appliance types.

## 5.2 Using a Multi-State Appliance Model

Having explored in Section 5.1 the performance of the power-state detection and clustering approach at extracting information from the whole-home power signal, the next stage of the implementation was to attempt disaggregation using appliance models, which can be created independently of any ground-truth measurements (but may also be created using them). An appliance model inspired by that used in [25] was implemented which contains a set of descriptive statistics related to the use profile of the appliance and the power states it generates.

The implementation allows for the creation of an appliance model in two ways, via the use of ground-truth measurements, or manually specified. An appliance model could also be built in 'real-time' by asking a user for feedback as appliances were turned off and on and their transitions detected, however, this has not been implemented and would be a priority in any further work.

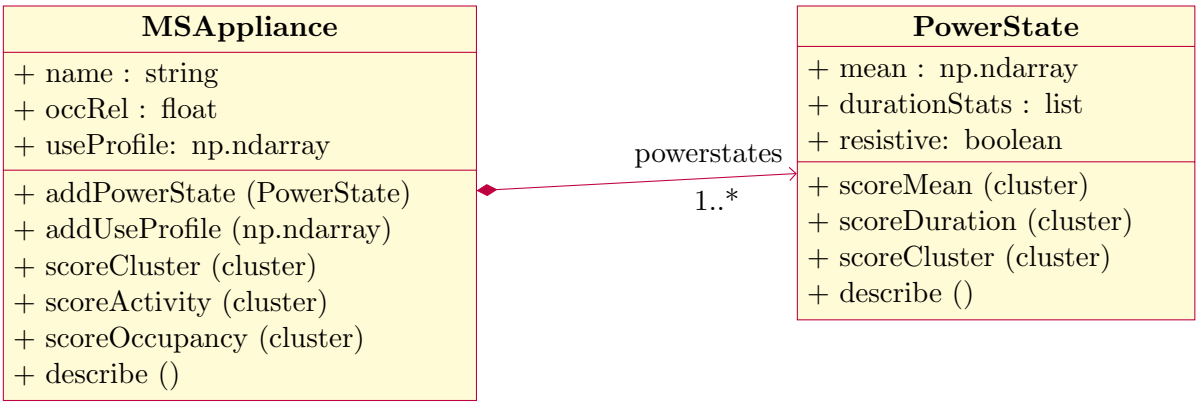
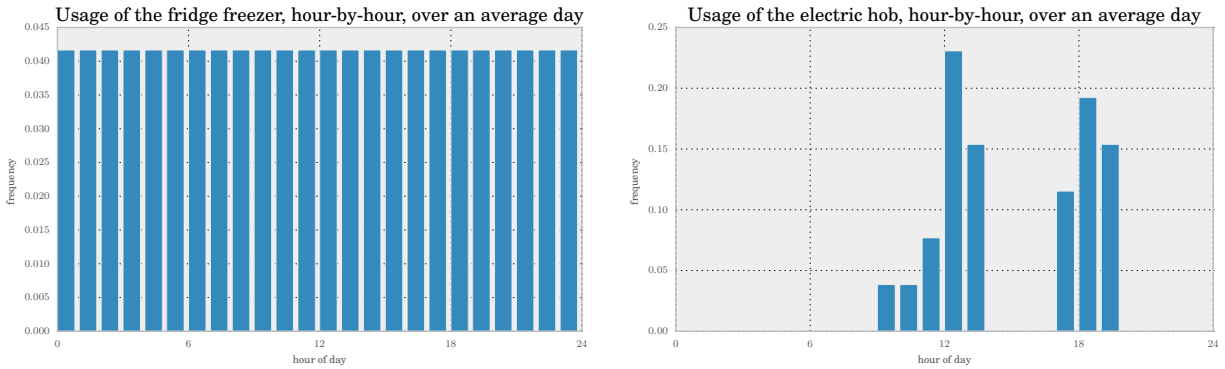



**Figure 5.5:** The MSAppliance and PowerState Classes

Figure 5.5 illustrates the MSAppliance class. It is composed of 1 or more PowerState classes. The MSAppliance is instantiated with a name, its use relative to the occupancy of the household and its use profile. The use profile comprises a histogram of 24 bins, each representing an hour of the day. It is normalised such that the area under the histogram is one, and represents the distribution of use that is expected of the appliance in each hour.

Figure 5.6 shows an example of the normalised use profiles of the fridge freezer within the household and the electric hob. One can clearly see that the fridge-freezer is equally likely to be on at any point throughout the day, whereas the electric hob is used around lunch and evening mealtimes.

### 5.2.1 Building the Appliance Models




**(a)** Fridge Freezer **(b)** Electric Hob

**Figure 5.6:** Use profiles of the household fridge-freezer and the electric hob

A procedure for building appliance models from ground-truth measurements was developed. The functions:

1. **Compute the Use Profile of the appliance**

   The time series data for the appliance is analysed, and a histogram built of the times of day that the appliance was in a detectable power state (above 70W). The histogram contains 24 bins, representing each hour of an average day.

2. **Use Mean-Shift clustering to extract Power-States**

   Mean-shift clustering is run on each appliance for which we have collected data. We use this to extract power-states occupied by the appliance above our noise-threshold of 70W (i.e, which we expect would be detected by our edge detector).

3. **Compute descriptive statistics for each Power-State**

   Each cluster of samples labelled by the clustering procedure is processed to extract descriptive statistics. We extract:

   - The centroid of the power-state.
   - The mean, median and standard deviation of the durations that the appliance was in that power-state.

4. **Post-process to remove Power-States with little data**

   We post-process the output of step 3 in order to remove those states in which the appliance spent less than 10% of its ON time or with few samples. This is done since the mean-shift clustering was found to produce occasional small clusters which did not represent steady-state power levels of an appliance.

5. **Remove Continuously Variable Appliances**

   We remove continuously variable appliances as we are aware from the discussion in Section 5.1.4 that we will be unable to detect these appliances with our edge-detection approach.

Figure 5.7 shows the centroid of the power-states extracted for each appliance by the above procedure. These are used to build MSAppliance objects, along with the other statistics computed during the procedure above.

It is worth remarking at this point on some conclusions that can be made from this graph, as they will be seen to affect our success at matching clustered data to appliances later in the section.

- **The Oven and Kettle produce an almost identical Power-State**

  Both of these devices can be seen to produce a power-state at $\sim$ 2800W. Worse still, they are both resistive devices, and thus will have a very similar power-factor. We thus expect that these devices will be clustered together.

- **The Hair-dryer and Toaster produce an almost identical Power-State**

  Again, these are both resistive devices, meaning that they will not be distinguished by their reactive power component in our signature-space.

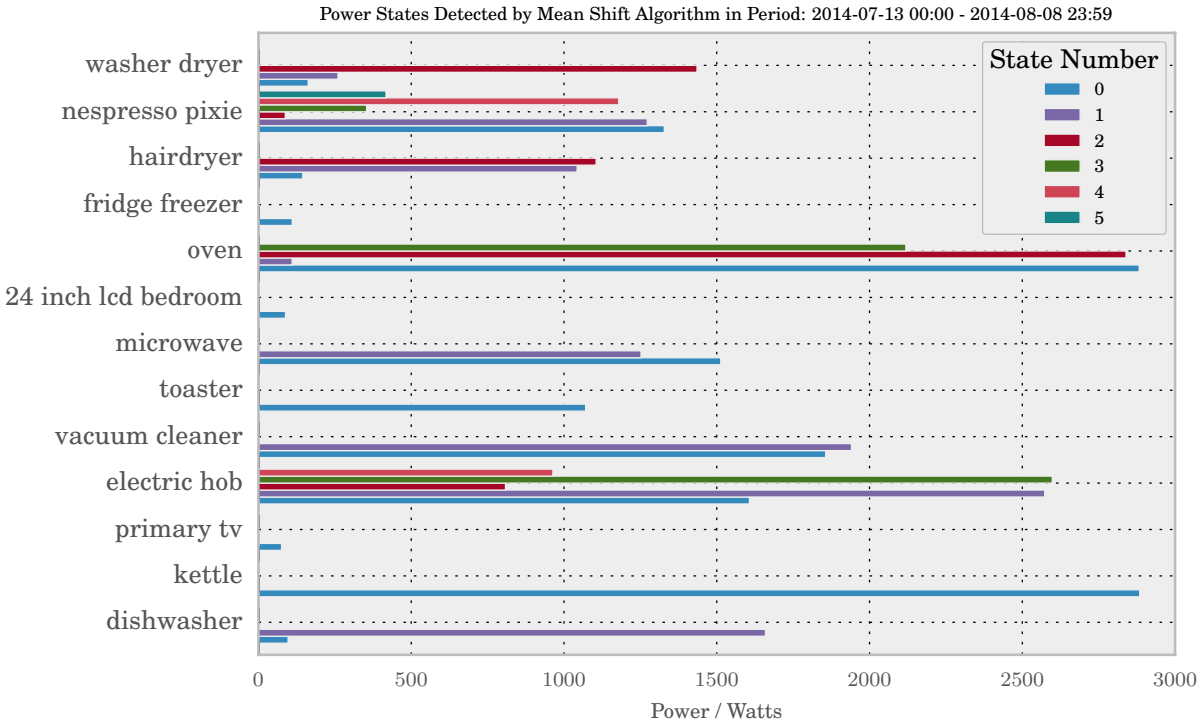



**Figure 5.7:** Power-States detected by mean-shift clustering of Appliance Ground Truth Data

### 5.2.2 Matching Appliance Models to Clusters

The approach taken to matching appliance models to clustered data was to find the nearest matching power-states by euclidean distance between the appliance's active/apparent power and cluster's active-power. For those 3 appliances with power-states nearest to the cluster, the similarity of the duration of the cluster's power-states was then assessed to those of the appliance and the histogram of the appliance and cluster's activity distribution compared. The similarity of the activity distribution is assessed by means of a chi-square test.

It should be kept in mind that in this assignment, we are not including any appliance models for lighting (as these must be manually created), and thus we expect energy to be over-assigned across our appliances.

The results of this assignment procedure are shown in Figure 5.8 and highlights some of the successes and key problems with the construction of our appliance models and the clustering of our data.

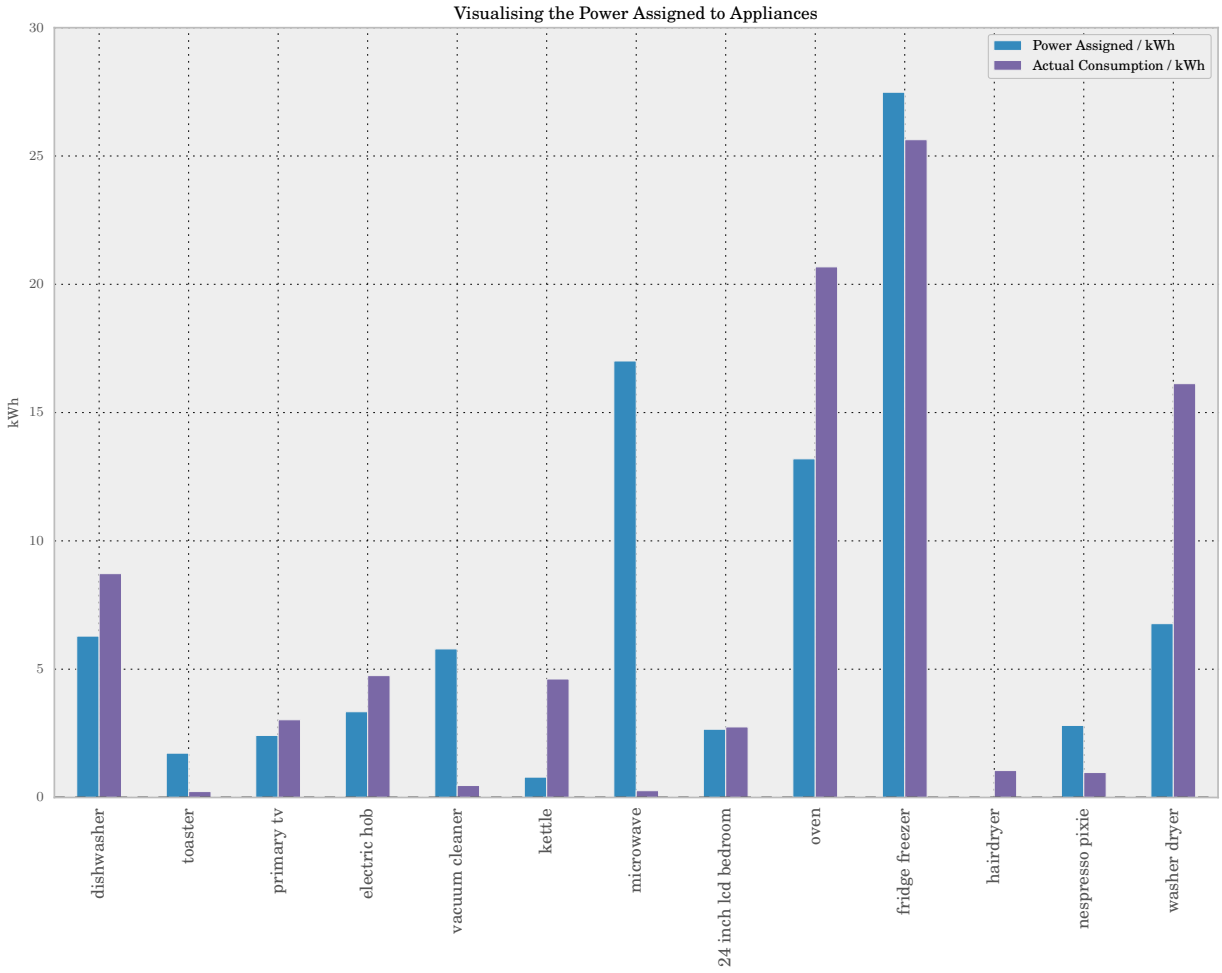



**Figure 5.8:** Power assigned to appliances using appliance models and mean power, duration mean and activity profile of power-states

- **Success in identifying states of fridge-freezer, electric hob, dishwasher, bedroom LCD, and primary TV**

  Four clusters are assigned to the electric hob, and account well for the energy it consumed. By reference to Figure 5.7 we see that the electric hob does not produce power-states that are in very close proximity with other devices. This has resulted in well formed clusters for the electric hob that do not contain power-states belonging to other devices.

  The fridge-freezer has had two clusters assigned to it. One accounts for 80% of the energy used, it does however underestimate it - likely as a result of the exclusion of energy used during transient state when the compressor starts. The second cluster likely belongs to light fittings containing 3 40W Halogen bulbs.

  The dishwasher was assigned one cluster, accounting for 72% of its energy consumption.

  The bedroom LCD was assigned one cluster, accounting for 95% of the energy consumed. The primary TV was assigned one cluster, accounting for 80% of its power consumption.

- **Failure to assign energy to kettle**

  The kettle is only assigned a fraction of its true energy consumption. It transpires that this is for the reason discussed in Section 5.2.1. The oven and kettle produce a purely resistive power-state at 2800W into which they are both clustered. We cannot overcome this problem without increasing the dimensionality of our signature space to include occupancy metrics, or state duration information for each sample.

- **Significant over-assignment to microwave**

The microwave has an enormous over-assignment of energy.

There are actually two clusters assigned to the microwave which account for 9.82kWh and 7.29kWh. The first actually seems to have been generated by the Nespresso Pixie (coffee maker). It seems likely that the sparse-sampling rate of the individual appliance monitor on this device has failed to account for a large amount of energy that it consumed. The device seems to rapidly switch on-and-off and this confuses both our edge-detector and means that the individual appliance monitor does not well capture its energy consumption.

The second incorrectly assigned cluster seems to have been produced by the washer dryer. Unfortunately the use of only apparent power measurements (as produced by the individual appliance monitor) to calculate the power-states of this device result in an under-estimate of the active-power of the power-state. This results in the microwave being found to be the closest matching device, and the mean duration of the power-state of the microwave and washer-dryer element appear to be sufficiently similar that even the comparison of these two statistics does not allow us to distinguish the state correctly.

- **Significant over-assignment to vacuum-cleaner**

  The vacuum-cleaner is assigned energy that should have been assigned to the oven. This occurs as a result of the combination of the use of apparent power measurements for this device and the mean-shift clustering performed on this data resulting in power-states corresponding poorly to those captured in the signature-space.

- **Assignment of multiple power states to Washer-Dryer**

  As a result of the washer-dryer's variable-speed motor, the clusters produced by our mean-shift clustering of this appliances data are very wide and the variable-speed drive on this device is really a continuously-variable device. The power-distribution is illustrated in Figure 5.9. This appliance has actually been assigned those clusters that were poorly fitted to other appliances, and which are located in the low-power region of our signature space.

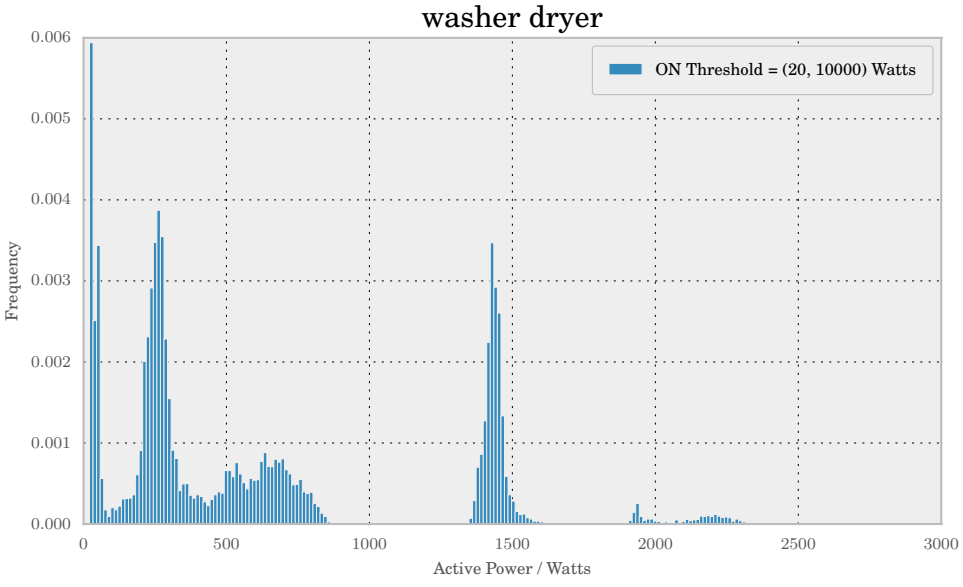



**Figure 5.9:** Power distribution of the Washer-Dryer

### 5.2.3 Adding Manually Specified Appliance Models

Since we do not have the ground-truth for light-fittings, testing was undertaken to see how well appliance-cluster matching would perform using hypothetical models of these appliances.

We create appliances that are shown as 'lighting 1' and 'lighting 2' in the Figure 5.11, and were designed to model halogen light fittings present in the home: one with 3 40W Halogen bulbs, and the other with 6 40W Halogen bulbs. The appliances are created using the code shown in Figure 5.10 (though this code has been simplified.

```
# Example of a use profile
aProfile = [0, 0, 0.0166, 0, 0, 0, 0.0166, 0.0166, 0.01667, 0,  0, 0, 0.0166,
      0.01667, 0.01667, 0, 0, 0.0166, 0.1, 0.233, 0.266, 0.216, 0.05, 0]

# Create an MSAppliance object and powerState within it.
lighting1 = MSAppliance(name='3 Halogen', powerState=PowerState(mean=[120,0],
                                duration[1200]) )
lighting2 = MSAppliance(name='6 Halogen', powerState=PowerState(mean=[240,0],
                                duration[1200]) )
# Add the use profile to the appliance
lighting1.addUseProfile(aProfile)
lighting2.addUseProfile(aProfile)
```

**Figure 5.10:** Simplified Python code extract illustrating the process of creating two new MSAppliance objects representing light fittings.

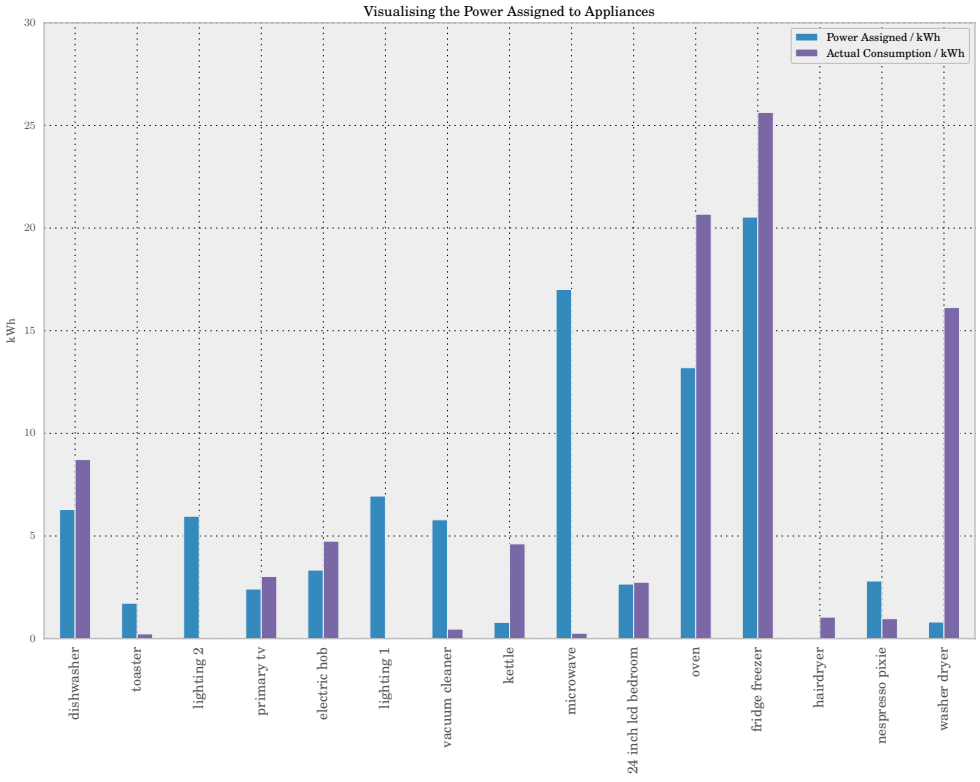



**Figure 5.11:** Power assigned to appliances using appliance models constructed from individual appliance monitors, and manually created lighting models.

Figure 5.11 shows the power assignment to individual appliances, including the two that were created by manually instantiating models of light fittings. The light-fitting models are the third and sixth bars from the left of the chart.

It is possibly to see that the two clusters that were erroneously assigned to the washer-dryer when the lighting models were not present are now assigned to 'lighting 1' and 'lighting 2' respectively.

This is a positive step, but it does show us that if we do not have a complete list of the appliances present in a home, this method of disaggregation will tend to assign energy incorrectly to the 'next best' match.

### 5.2.4 Summary

We have seen in this section that appliance models were developed to allow for assignment of clustered data to real world devices. These models constitute a class representing the appliance as a whole, the MSAppliance, which is composed of any number of objects of the PowerState class.

We may construct an appliance model from either ground-truth measurements from an individual appliance monitor, or we may construct a model from parameters that we can observe in the real world (such as the quoted power consumption of a device, the time of use of a device, and the duration of use of each device).

Some simple statistics extracted from the clusters and individually monitored appliances are used to compare one with the other, and the best matching appliance under this regime is used assigned the cluster as 'belonging' to it.

It is has been seen that there are some significant weaknesses with this approach. We do not have enough information in some cases to distinguish between two devices that occupy very similar power-states (as was seen with the kettle and oven-element). In other cases, building the appliance models from data collected by apparent power measuring appliance monitors was seen to result in poor correspondence to devices which were not resistive, as with the washer-dryer and microwave.

It has been seen that devices (such as the washer-dryer and nespresso machine) which have very short lived power states can be represented significantly differently when monitored at 1s intervals vs the 6s interval of the individual appliance monitor. It appears that the sparse-sampling monitors do not capture every activation.

It has also been seen that the edge-detector which is expecting steady-states can fail to detect very short lived states, even with a 1Hz sampling rate. This was discussed in Section 3.3.

We have seen that the method has some success in allocating energy to appliances with well separated clusters in the signature space - the fridge-freezer, the electric hob, the dishwasher, and several displays in the home. This also seems to apply to the light fittings when included in the list of appliance models.

# Chapter 6

# Evaluating Temperature Data and Occupancy Data

## 6.1 Power consumption of appliances at varying temperatures

Temperature information was used to compare the energy consumption of various household appliances against the internal temperatures in the hopes that this might be used to aid disaggregation performance.

Temperature measurements taken at 1-minute intervals were down-sampled to daily records and the average, minimum and maximum temperature recorded for each day.

Total energy consumption data for each device during each day was computed by summing the recorded power use at each 6 second sample over the 24 hour period.

Table 6.1 shows the results of this analysis for the period 13/07/2014 - 08/08/2014. We see some noteworthy results within this table. The vacuum cleaner was only used twice during the period, and thus has unreliable measurements for correlation. We find that the fridge's power consumption is reasonably well correlated to temperature. This is as we expect - temperature is one of the factors which dictate how often the refrigerator must run its compressor in order to maintain a cool internal temperature. As was discussed earlier, when the fridge is loaded with fresh consumables we expect that its power consumption will rise more significantly. In fact, consulting Figure 6.1 (which shows the minimum , maximum and average temperatures on each day for a month long period along with the total energy consumption of the fridge) we can determine the two occasions on which the refrigerator was loaded with goods - they show as dramatic spikes in the energy consumed. This gives us some insight into the privacy concerns of a disaggregation system - if we can accurately capture the power states belonging to the fridge-freezer, we can tell when the householder does their shopping!

| Device | Pearson Correlation Coefficient (Internal) | Pearson Correlation Coefficient (External) |
|---|---|---|
| vacuum_cleaner | 0.942 | -0.960 |
| fridge_freezer | 0.523 | 0.478 |
| nespresso_pixie | 0.361 | 0.042 |
| atom_pc | 0.284 | 0.448 |
| stereo_speakers_bedroom | 0.196 | 0.257 |
| dishwasher | 0.184 | 0.102 |
| washer_dryer | 0.155 | 0.026 |
| i7_desktop | 0.056 | 0.339 |
| core2_server | 0.044 | -0.073 |
| steam_iron | 0.005 | -0.215 |
| hairdryer | -0.052 | -0.074 |
| treadmill | -0.095 | -0.355 |
| oven | -0.120 | -0.216 |
| network_attached_storage | -0.121 | 0.025 |
| 24_inch_lcd_bedroom | -0.145 | -0.007 |
| sky_hd_box | -0.156 | -0.231 |
| home_theatre_amp | -0.184 | -0.218 |
| microwave | -0.263 | -0.226 |
| electric_hob | -0.293 | -0.209 |
| primary_tv | -0.294 | -0.119 |
| 24_inch_lcd | -0.323 | 0.034 |
| kettle | -0.334 | -0.356 |
| toaster | -0.351 | 0.101 |

**Table 6.1:** Pearson product-moment correlation coefficients for appliance energy consumption against internal and external average temperature on a daily basis for period 13/07/2014 - 08/08/2014.

We also see from this table that many of the devices within the home do not show a strongly positive or negative correlation to temperature (either external or internal).

It is interesting to see that the kettle and toaster have a reasonable negative correlation between

temperature and their energy consumed. It seems that these devices are less likely to be used on warm day!

It seems from the data that has been collected that temperature may give us some useful information to use in disaggregating power states. However, it is noteworthy that those appliances which do seem to have reasonably strong correlations with temperature were some of those that were more successfully identified using multi-state appliance models not incorporating temperature. These include the toaster, fridge-freezer, primary TV, and electric hob. The kettle is another device that has a moderate negative correlation with temperature. As was discussed in Section 5.2 whilst the edge-detection algorithm was able to detect the power-states of this appliance, they were indistinguishable in the signature space from those of the oven's heating element.

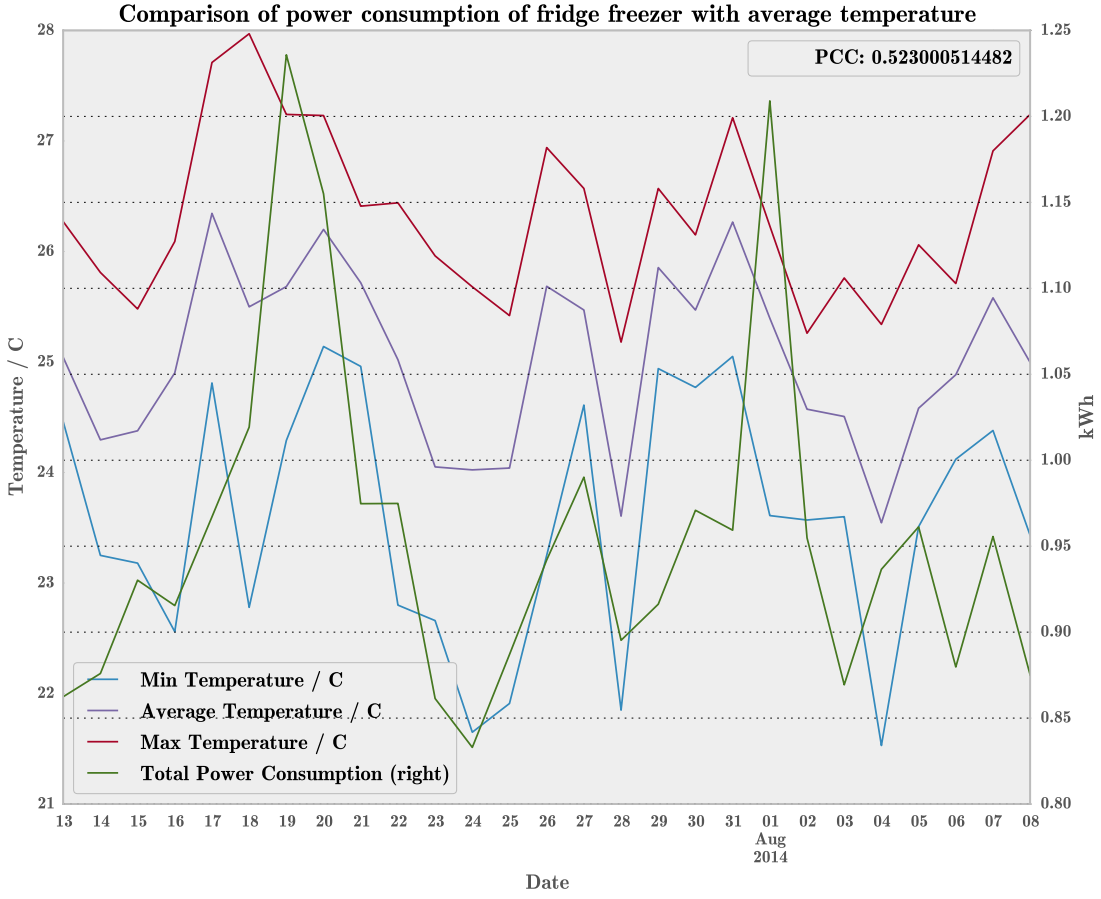



**Figure 6.1:** Comparing energy consumption of fridge-freezer with average temperature over period 13/07/2014 - 08-08-2014. The two large peaks in energy use indicate days on which the refrigerator was loaded with shopping.

## 6.2 Attempts to compare appliance signatures at different temperatures

Appliances whose signatures were expected to vary with external temperature were investigated. The dishwasher and washing machine must both heat water to a desired temperature in order to complete their cycles, so these devices were candidates for investigation.

Profiles for the dishwasher and washer-dryer can be found in Appendix D. Unfortunately, there was little variation in total energy consumption of these devices over the range of temperatures that were experienced during the monitoring period. It is also the case that comparison of the energy

use of these devices is complicated by the fact that they seem to use internal sensors to detect the quantity of the load and adjust their behaviour as a result. Without a 'standard load' with which to test these machines, comparison between them is fraught with uncertainty.

The rapidly cycling dryer element in the washer-dryer is also unlikely to be accurately quantified in energy consumption by the individual appliance monitor that was used. Uncertainty is also compounded by the fact that an apparent power CT clamp monitor was used, which does not have insight into the mains voltage present at the time.

Ultimately, there was just not enough good data to make a meaningful comparison.

## 6.3 Use of appliances relative to household occupancy

Following investigation into the energy consumed by appliances across a range of temperatures, appliances were assessed for their relative use when the household was occupied, and when it was not occupied.

Occupancy data is collected approximately 3 times per minute per device. Mobile devices are polled for their presence as often as possible, but the process is delayed if one or both devices are not present, as the polling software has to wait for a 'time-out' to occur. Occupancy data can also fail to detect a device and report a false negative if there is interference present or the signal is weak. This is usually reflected by an occasional sample being saved which reports a device as 'away' amongst many samples which report that it is present. False positives are not possible. We accept that if *either* occupant's device is present then the household is occupied.

We have seen that appliances are generally sampled every 6 seconds by our individual appliance monitors, and also suffer from occasional drop-outs due to interference or collision of packets.

In order to align this data, both the appliance measurements and occupancy measurements were re-sampled to 5-minute intervals.

The re-sampling method for the occupancy data was to seek *any* report of the presence of a device within the 5-minute interval, and record the occupant as present for that interval. Since the occupancy data does not report false-positives, we take any five minute period in which all samples were False to indicate that the occupant was not present.

Individual appliance monitor data is first processed to screen devices against a low-power threshold and high-power threshold. This removes any 'vampire' measurements from the appliance's samples and also any erroneously high measurements that the meters occasionally report. Individual appliance monitor data was re-sampled using the maximum power value detected in any five minute period.

The data for each appliance and the household occupancy was then aligned on the time-stamps for each sample. A boolean test is then performed on the appliance level data: if the power use in the period is greater than the ON threshold for that appliance, record it as ON (True) otherwise record it as OFF (False).

Figure 6.2 shows the number of samples recorded for each appliance which showed it to be ON for periods when an occupant was home against those periods where no occupant was present.

Five devices show the same number of samples for both an occupied household and unoccupied household. These are the always on devices: the data logging atom-pc, the core2 server, the home-theatre amp, the network-attached storage device, and the sky-hd box. The comparison in number of samples collected for each of these devices when occupied and empty shows the ratio of occupancy/non-occupancy of the household.

The fridge-freezer can be seen to be on slightly less than half the total time, and the ratio of its behaviour when the household is occupied/unoccupied is, as expected, the same as that of those devices that are always on. Its behaviour does not, in general, change with occupancy.

We can see from the figure that several devices do not appear at all (or have a very small number of samples assigned) when the household is not occupied. These include the devices one would expect: the treadmill, oven, televisions, kettle and other cooking devices.

We can consider further the behaviour of appliances relative to occupancy by consulting Table 6.2. This table shows the number of 5-minute periods (also referred to as samples) during which

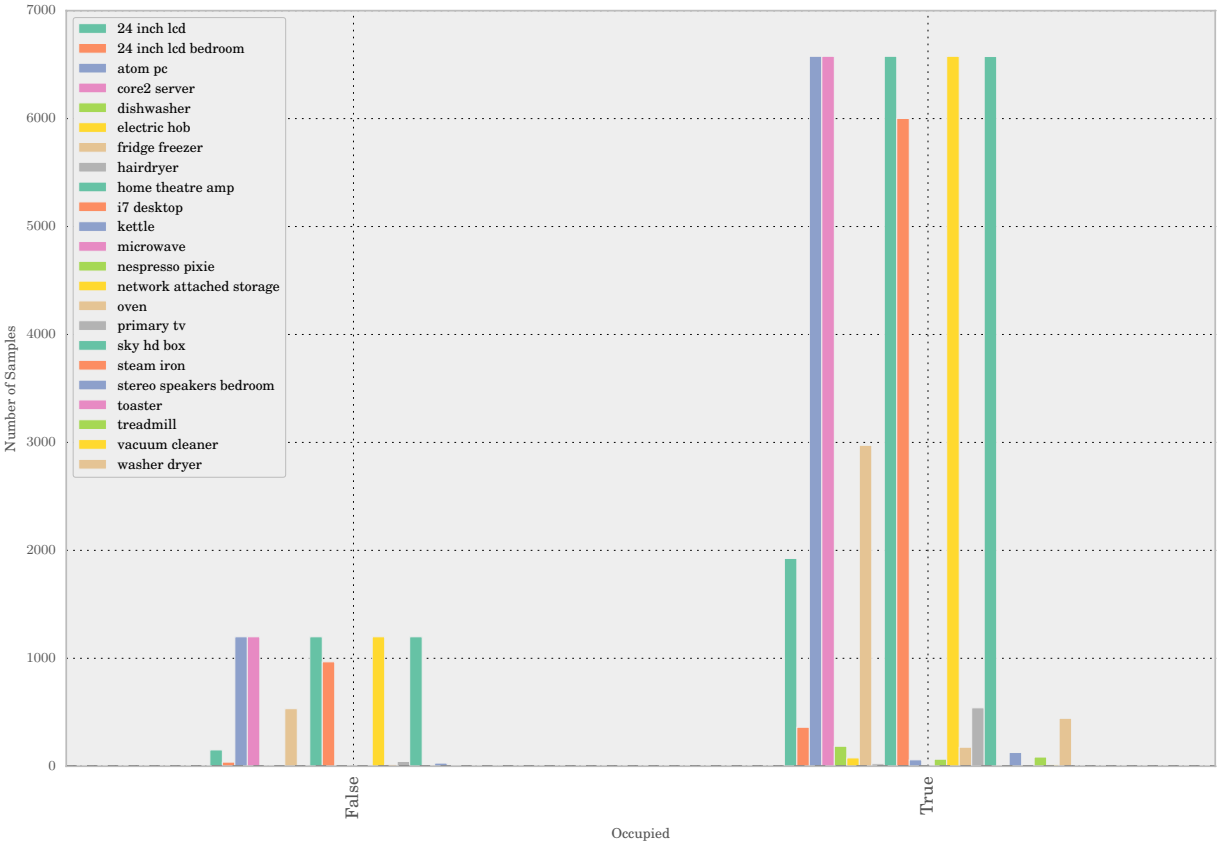



**Figure 6.2:** Comparing the use of appliances in the household with household occupancy. The y-axis shows the number of periods in which the appliances were 'ON'. Colour coding from the legend runs from left-right along the x-axis.

the household was occupied, and the number of periods during which it was unoccupied.

The table also contains the details of each appliance for which sub-metered data was available, and shows the number of periods during which the appliance was seen to be on when the household was both occupied and unoccupied.

We may draw some interesting conclusions from this data:

- **Large relative error on sparsely used devices.**

  Devices such as the kettle (which clearly can only be used when there is a occupant at home) are activated for relatively few 5-minute periods during the month. The kettle for instance was seen to be active 68 times during the monitoring period. It seems that on 9 of these occasions, no occupant was detected. This may be due to a phone's battery dying, or perhaps the kettle (or another appliance used with it) produces interference that affects the blue-tooth signal of the phone. Another possibility is that the location of the kettle experiences a low signal from the polling device. In any case, whilst the kettle should effectively have an 'inf' ratio by virtue of its never being used when there is no occupant, in fact we see that it experiences a ratio not far that of the fundamental occupancy ratio of 5.485.

  A similar issue is seen with the steam iron, which was active very little during the month.

- **The household is occupied 85% of the time.**

- **Some expected devices are only ON when an occupant is present.**

  The treadmill, vacuum cleaner and hair-dryer are all recorded as only being used when an occupant is present.

| Item | Occupied Periods | Non-Occupied Periods | Occuped / Non-Occupied Ratio |
|---|---|---|---|
| *Household* | *6577* | *1199* | *5.485* |

| Item | Occupied ON Samples | Non-Occupied ON Samples | Occuped / Non-Occupied Ratio |
|---|---|---|---|
| 24 inch lcd | 1925 | 151 | 12.748 |
| 24 inch lcd bedroom | 361 | 38 | 9.5 |
| atom pc | 6577 | 1199 | 5.485 |
| core2 server | 6577 | 1199 | 5.485 |
| dishwasher | 185 | 1 | 185 |
| electric hob | 77 | 5 | 15.4 |
| fridge freezer | 2973 | 533 | 5.578 |
| hairdryer | 23 | 0 | inf |
| home theatre amp | 6577 | 1199 | 5.485 |
| i7 desktop | 6001 | 967 | 6.206 |
| kettle | 59 | 9 | 6.556 |
| microwave | 9 | 1 | 9 |
| nespresso pixie | 65 | 9 | 7.222 |
| network attached storage | 6577 | 1199 | 5.485 |
| oven | 176 | 17 | 10.353 |
| primary tv | 541 | 44 | 12.295 |
| sky hd box | 6575 | 1199 | 5.484 |
| steam iron | 4 | 1 | 4 |
| stereo speakers bedroom | 127 | 29 | 4.379 |
| toaster | 9 | 4 | 2.25 |
| treadmill | 85 | 0 | inf |
| vacuum cleaner | 12 | 0 | inf |
| washer dryer | 444 | 6 | 74 |

**Table 6.2:** Tabulating the number of 5-minute periods during the period 2014-07-13 - 2014-08-08 in which each appliance which was sub-metered was active, and whether the household was occupied at that period.

## 6.4 Summary

Occupancy data certainly seems to contain useful information to enable the assignment of probability of use to an appliance given the presence (or not) of occupants in the household.

However, when consideration was given to how best to implement this data in mapping clusters to appliance-models, it became apparent that the nature of appliances in the household diminish the value of occupancy data.

Consider those appliances which are only, or enormously more likely to be used when an occupant is present: generally, the high-powered kitchen appliances, televisions and other displays, and lights. We have seen that the household television was identified rather well in our energy assignment, and that it seems that the same would apply for the light fittings (when appliance models for these devices were created). The oven, electric hob and kettle are other candidate appliances that we expect to only be used when an occupant is present. We have seen that the electric-hob's energy consumption was disaggregated well - but that the kettle and oven have an identical resistive power state which prevented us from splitting the energy used between these appliances. Knowing that an occupant is present does not allow us to 'disentangle' the power-states captured in this cluster in our signature space.

Temperature data may allow also allow us to modify the probability that power-state belonged to one device or another. In this case, since the kettle seems to be negatively correlated with temperature, we might be able to 'tag' each power state with the temperature before it was inserted into the cluster, and then probabilistically assign each one to a particular device.

The issue with this approach is that it runs contrary to the model of a cluster as representing one power-state of one device, but as we have seen, this is not necessarily valid and has hampered our ability to assign power to appliances.

# Chapter 7

# Conclusions and Further Work

## 7.1 Evaluation of Disaggregation Accuracy

Table 7.1 shows the results of disaggregation using the optimal clustering parameters determined in Section 4.

I am happy with the performance of the system in disaggregating the fridge-freezer and dish-washer, and somewhat surprised at its ability to assign energy to the 24 inch LCD in the bedroom. This device produces a cluster centred at 84W active power - not far above our noise threshold of 70W. At the outset of this project, I had expected that the implementation would be far more successful in assigning energy to devices at higher power levels. It is clear that this is not necessarily the case. However, in the best case, assigning only about 40% of energy correctly cannot be considered a great success.

| **Algorithm** | **Parameters** | **Clusters Formed** | **MSC** | **Fraction Energy Assigned Correctly** |
|---|---|---|---|---|
| K-means | $K = 27$ | 22 (after post-processing) | 0.659 | 0.409 |
| Mean-Shift | $Q = 0.029$ | 22 (after post-processing) | 0.66 | 0.41 |
| DBSCAN | $minPts = 11$, $N_{Eps} = 8$ | 19 | 0.502 | 0.323 |
| Hart | $minPts = 10$ | 25 | 0.6 | 0.415 |

**Table 7.1:** Comparing the parameters and fraction of energy assigned correctly for four algorithms.

We see that K-means, Mean-Shift and Hart clustering all achieve very similar results. This is likely due to the nature of their cluster models and the optimisations performed for our data. It is interesting to see that the Hart method has the highest fraction of energy assigned correctly on our dataset (although only by a very small margin). It competes well with two very widely used clustering algorithms and can be run dynamically.

It is worth noting that because DBSCAN creates arbitrarily shaped clusters it tends to result in lower Mean-Silhouette Coefficients (MSC) when its output is assessed [49]. We also see that DBSCAN has the worst performance overall. This can be attributed to its classification of large numbers of samples as noise. Even post-processing the clustering to assign 'noise' as outliers of clusters, DBSCAN does not perform well.

As was discussed in Section 4, there is value in clustering 'local' regions, even if the density of points is very low. The failure of DBSCAN to do this is a feature of its relative unsuitability to our problem domain.

Any future approach using an edge detection technique would have to be built upon the expectation that clusters of power-states are not likely to be representative of one device alone. There are simply too many devices with very similar power-states for this assumption to hold. It has been the major limiting factor in disaggregation accuracy.

## 7.2 Limitations

- **Failure to disaggregate resistive appliances with similar power-states.**

  The disaggregation algorithm fails to disaggregate the kettle and oven grill element. It instead models these two appliances as one cluster, and cannot attribute the energy across two devices. A similar issue is experienced with the toaster and hair-dryer, which have resistive elements that draw the same power. This is a fundamental issue in modelling each cluster as belonging to one power-state of one appliance. Primarily resistive devices will not be separated in the reactive axis of our signature space, and so if they draw the same power, will be clustered together.

  We cannot remedy this issue without altering our cluster model to encompass the possibility that multiple distributions of power-states are contained within a cluster.

- **Lack of accurate energy consumption data on fast-cycling appliances.**

  It appears that the data collected for the coffee maker (nespresso pixie) may significantly underestimate the actual power consumption of this device, and thus it is difficult to assess the success of our identification of states belonging to this appliance. The individual appliance meters send on sample approximately every 6 seconds but this seems to be the instantaneous power. Thus, if a device is powered up for 5 of the 6 seconds in the interval but has stopped drawing power at the point the sample is sent, this energy is unaccounted for.

- **Difficulty in identifying edges for short-lived power states.**

  As was seen in Figure 3.4a, the edge detection algorithm attempts to detect steady-state power-states and 'ride over' transients. It appears from monitoring the household that devices which cycle power-states at rapid intervals (such as the coffee maker) are not well accounted for by the edge-detection process.

- **Impossible to account for 'vampire load' or variable state appliances.**

  The edge-detection approach cannot disaggregate or account for 'vampire' power drawn by appliances that are in standby or always on. This was seen in the case of the computers within the author's household, despite their consumption of over a quarter of the total energy consumed within the month, their power-states were not detected. Variable state appliances, such as dimmer switches, would also not be identified by this approach, and in fact in testing on the UKDALE dataset, significantly impact the ability to form clusters at lower power levels in the signature space due to their addition of 'noise' samples to the region.

- **Possible underestimate of power for fridge-freezer (and other appliances).**

  The avoidance of the transient state of the fridge-freezer and averaging seems to result in an underestimate of the power-use of such devices in general. Any device with a start-up transient will be underestimated.

  However, it seems likely that the use of CT type monitors with only 'estimated' apparent power data produced (since these monitors cannot know the current line voltage) makes it difficult to be certain of our disaggregation accuracy. The mismatch between the power-states observed by these monitors and those seen by the calibrated whole-house monitor (which has access to voltage information) causes a great degree of discrepancy in the total power consumption detected by the whole-house monitor and the individual appliance monitor.

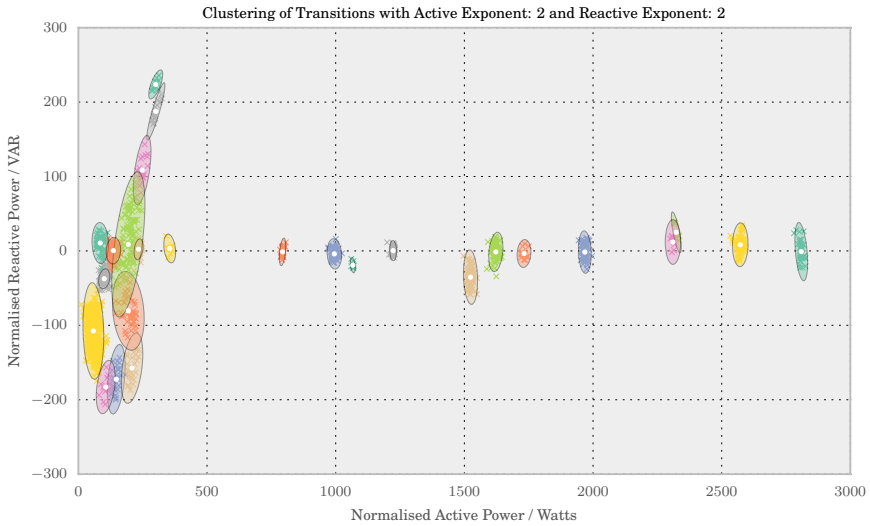
- **'One Cluster - One Appliance' model fundamentally flawed.**

  It is clear that since George Hart's original work on a similar NILM system that the power consumption of many devices has reduced. It is notable that the power demand of the household fridge-freezer in the author's home is less than half that of the refrigerator that was recorded in Hart's home in 1985 [25]. However, the increases in efficiency that have been made in household appliances have tended to crowd the region of of the signature space

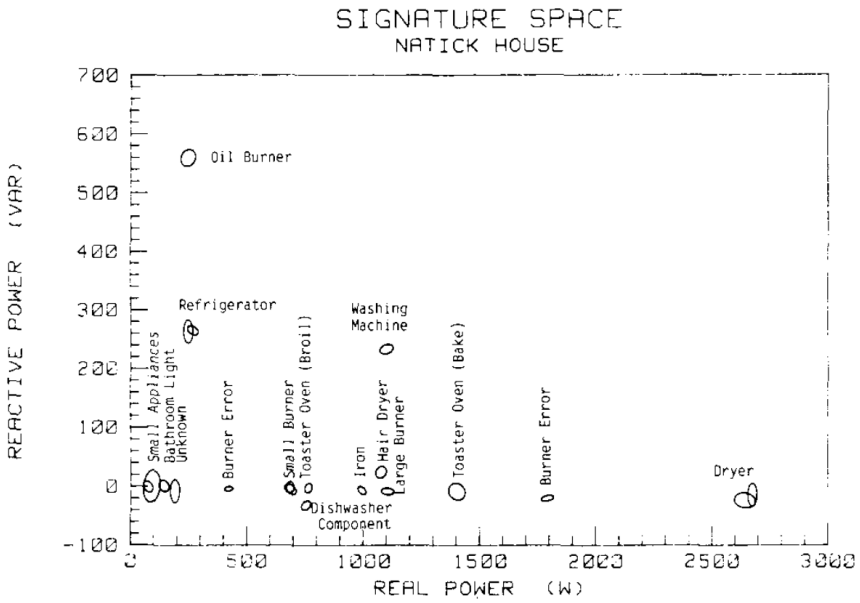
between 50 and 100W active power. Figure 7.1 compares the signature space in the author's home with that give in Hart's 1989 paper [26]. It is clear to see that the signature space at the low power range is far more crowded. It is also likely the proliferation of switched-mode power supplies and increasing commonality of computers has resulted in more devices drawing power in this area of the signature space.

The drive to increase efficiency of appliances seems to result in a general 'compression' of the signatures within the low power area of the signature space. This has undermined the assumption that each cluster will only contain states produced by one appliance.

The EU Energy Efficiency Directive 2012/27/EU [19] is likely to further increase the number of appliances in the 0-2000W active power range as mandatory bans on the sale of high-power goods such as kettles and vacuum cleaners come into effect.




**(a)** Complex power plane in author's home. In this plot, the sensor offset has been corrected for ease of comparison with 7.1b.




**(b)** 'Regions of the complex power plane associated with appliances' (Hart, George W,1989:p.13)

**Figure 7.1:** Comparison of author's household signature space with that given in [26]

## 7.3 Further Work

### 7.3.1 Incorporate occupancy data into disaggregation

Occupancy data could provide valuable information into the likely appliance to have produced a power-state. It might be possible to use the *transition* of the household from a non-occupied state to an occupied one to increase the likelihood of a 100W transition shortly thereafter being assigned to a light, rather than the refrigerator.

### 7.3.2 Improve Multi-State Appliance Models

The appliance models that have been developed are rather crude. We make use of limited statistics in characterising the power states of an appliance (including the mean of the power demand, and mean and standard deviation of the duration of each ON state of the device). It would be beneficial to actually characterise the nature of the distribution of durations for each appliance, for instance, the kettle may have an exponential distribution of durations.

One issue that was experienced when experimenting with improving the models was the variation in number of samples across differing appliances. The fridge-freezer for instance has hundreds of activations over a month. The vacuum cleaner and steam iron had few, if any. This made it difficult to apply the same method across all appliances.

In a similar vein, some clusters in the signature space are very dense and thus it is possible to try and extract more information to parametrise the distribution of samples within them. Others are very sparse. The differing density of clusters may actually be a useful metric in itself!

### 7.3.3 Attempt to discover multiple distributions within individual clusters

Given a sufficient density of samples within a cluster, it should be possible to use the duration of each of the power-states to assess whether there were in fact multiple 'peaks' in the distribution of durations. The presence of multiple peaks could indicate that whilst the power-states occupy the same area of the signature space, they were produced by two different devices (or one device but in a different mode of operation). This might allow us to separate the kettle from the oven grill element.

### 7.3.4 Remove deprecated code and increase efficiency

The project was undertaken in an iterative manner, with each portion of the disaggregation pipeline built and tested in turn. This approach, combined with my unfamiliarity with Python at the start of the process made it difficult to plan exactly what type of structure and information to pass on to the next stage in the pipeline at each point. On reflection, it would be best if certain extra data (such as occupancy status) was passed into the clustering phase of the pipeline.

There are a number of inefficiencies within the architecture which require the retrieval of information from an earlier stage in the pipeline later in the disaggregation process. This results in high memory usage and reduces the performance of the system. Rather than having to store all of the information relating to transitions throughout the entire process, I now realise that certain salient information could be extracted earlier in the pipeline (such as after pairing) and then the rest of the data removed from memory.

There are also multiple points in the system where parallelism could be employed to speed up processing.

### 7.3.5 Separate edge-detection, pairing and clustering into three processes

The edge-detection, pairing and clustering processes run one after the other. It would be preferable and faster to run each separately and use inter-process communication to pass data between them. This would restrict us to the use of the dynamic Hart clustering method (but this gave the

best performance in any case). This separation would be required in order to implement a 'live' disaggregation system.

### 7.3.6 Build an open database of appliances

Any 'practical' disaggregation system to be built on the edge-detection principle requires a database of appliances and the power-states that they occupy, and that we might detect. It would be beneficial for anyone that wished to disaggregate their household power data to have access to a database of appliance information from which their monitoring system might look up matches for the devices in their household, without ground-truth measurements being required.

### 7.3.7 Implement a 'live' disaggregation tool that could accept user feedback

Implementing a tool which would detect power-state transitions in real-time would allow for some truly interesting applications and enable immediate user feedback. This feedback could be used to inform on what device had been activated to produce the power-state. This would enable gradual learning of the appliances that produced each cluster, and if two appliances (such as the kettle and oven grill) produced a similar power state, the algorithm could more frequently ask a user for information on the device in use at each activation. This may enable automated separation of the appliances. For instance, an activation of an appliance at 7am would be most likely to a kettle and the user may not be asked for feedback. However, an activation of the power-state at lunch-time might be ambiguous, and an email or 'push message' might be sent to the householder asking for confirmation of which device was activated.

It would be desirable to build a web-interface to the NILM to allow the user to see their instantaneous power consumption, and perhaps which devices were thought to be active. If the NILM had made errors, the user could then correct which devices were active and aid in the learning of the system.

A web service might also enable the user to easily submit profiles of their appliances to the open database discussed above.

## 7.4 Conclusion

We may look back to the original aims of the project in assessing what has been achieved. An open-source implementation of a steady-state detecting disaggregation system has been produced. Nipun Batra has been able to incorporate the steady-state detection and pairing algorithms into NILMTK as of the time of writing.

How well does it work? The system has been successful in disaggregating some of the energy consumed by certain devices within the author's household. It was most effective in disaggregating the fridge-freezer and dishwasher, but it seems to be easily confused by the crowded signature-space in the region up to 500W active power. It fails to assign energy correctly to devices which rapidly switch power-states (such as the coffee maker and washer dryer). Jack Kelly experienced similar difficulty in identifying the power-states of his tumble dryer in [35], even using a differing 'signature' based approach. It seems that these devices pose a difficulty for NILM algorithms. The approach also makes it impossible to disaggregate the energy used by differing appliances which produce very similar resistive power states.

The extensions of the method to include occupancy and temperature data were not implemented due to time constraints. This was primarily as a result of the discovery that the 'one-cluster one-appliance' assumption was invalid. Data was collected within the household however, and an analysis shows that it may indeed by useful in informing future disaggregation algorithms. It seems likely that the approach used would need to be significantly modified, quite early in the pipeline, to allow this information to be incorporated.

We have established that in general, the power consumed by devices in 2014 is less than it was in 1985 when this approach was conceived. It seems that this has served to undermine the 'one-cluster one-appliance' model that was used in the earlier work. Thus, whilst the edge detection

and pairing approach is clearly able to extract information from an aggregate power demand and the clustering process produces well formed clusters - modelling these as belonging to only one appliance has been seen to be inappropriate in the general case.

Has the project then achieved its original aims? An open-source implementation of the Hart edge-detection NILM algorithm has been produced and tested on real-world data, and could be used as a benchmark. Thus, whilst the proposed extensions were not implemented, the project has achieved its primary aims.

# Appendix A

## User Guide

The system described in this report is implemented in a number of Python modules which have been tested to function on an x86-64 Linux PC, though the software should function on other platforms.

Development was conducted using the IPython notebook environment. Included with the source code is a folder named `/notebooks` which contains a variety of annotated notebooks that illustrate the operation of the disaggregation system and enable the production of graphs and charts at appropriate points to aid in understanding.

In order to access these notebooks, IPython should be installed and a notebook server activated from within the `/notebooks` folder using the following command: `ipython notebook`. A web-browser should open with a list of the available notebooks, which can then be explored by the user.

The notebooks and modules provide importers for the UKDALE dataset as well as the author's 'Kinetica' dataset. Due to the size of these datasets, they have not been included in the archive. The UKDALE dataset is available from Imperial's DOC servers. The Kinetica dataset can be provided on request.

## A.1 Software Dependencies

Most of the dependencies below are available in the Anaconda Scientific Computing Distribution for Python.

- `Python 2.7`
- `numpy` and scipy
- `scikit-learn`
- `pandas 0.14`
- `statsmodels`
- `nilmtk v0.1.1`
- `matplotlib`
- `prettyplotlib`

# Appendix B

## Clustering

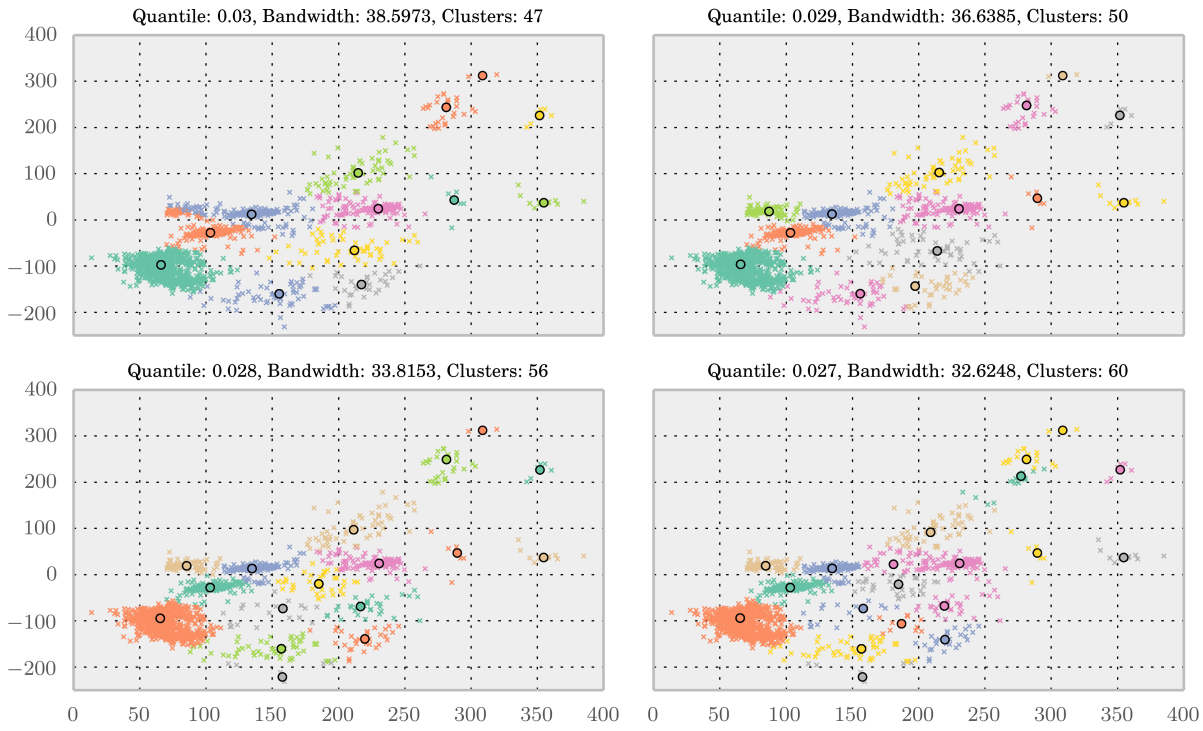



**Figure B.1:** Visualising the outcome of mean shift clustering for very narrow Quantile Values. Black circles represent centroids, whilst ×'s are individual samples. The samples are colour-coded to the centroid.

# Appendix C

## Python Class Diagrams

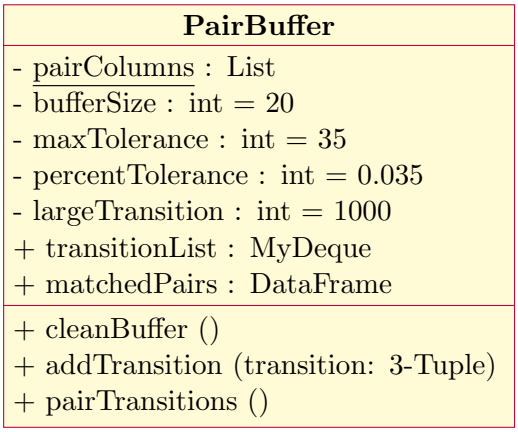



**Figure C.1:** The PairBuffer Class

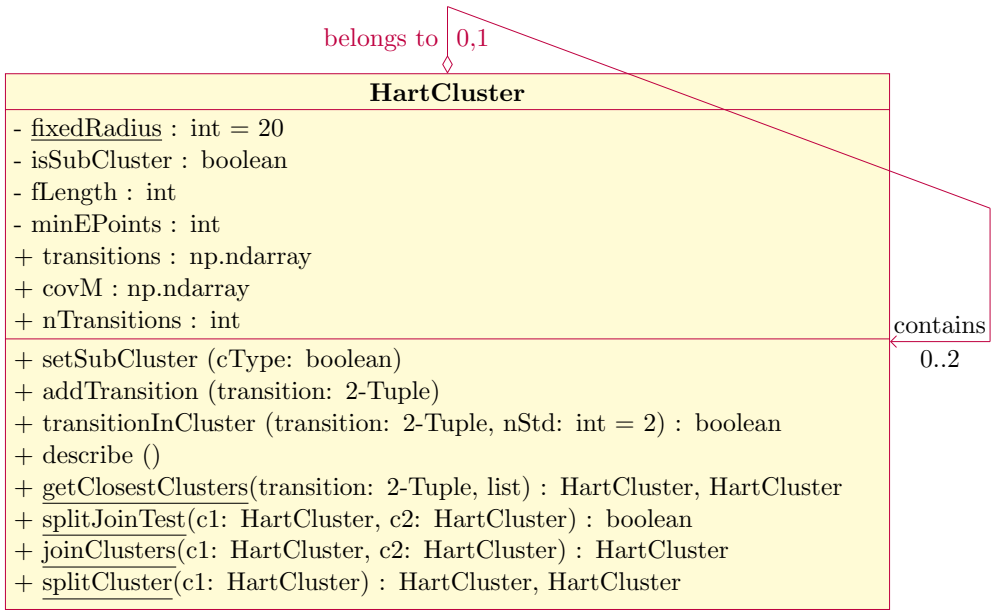



**Figure C.2:** The HartCluster Class

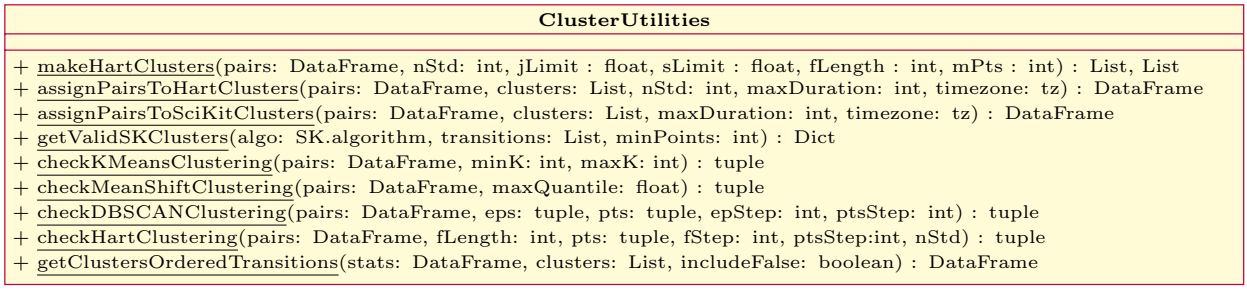
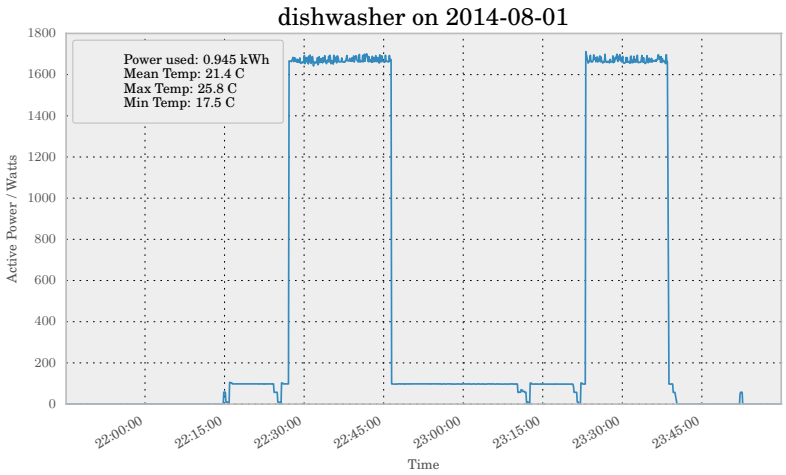


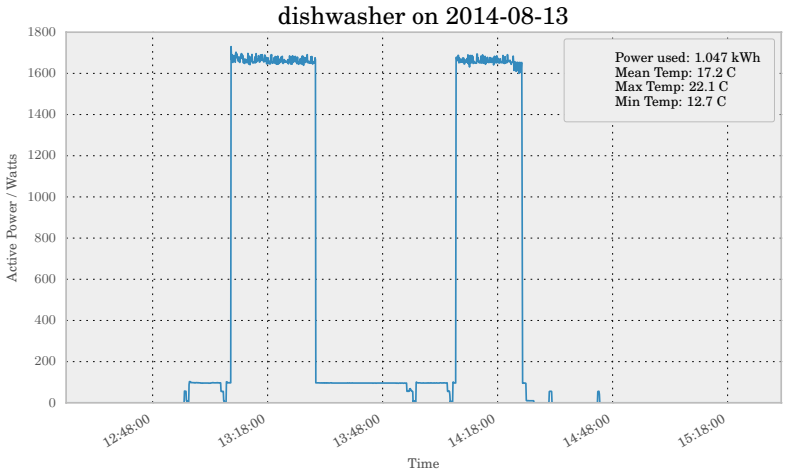

**Figure C.3:** The Clustering Utilities Class

# Appendix D

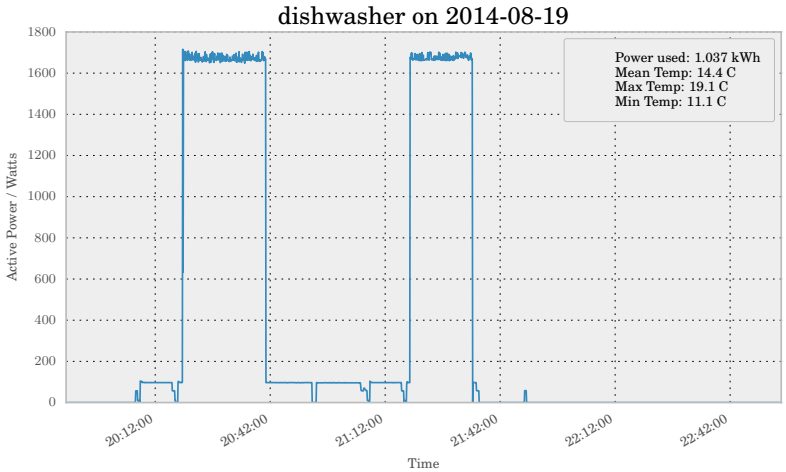
## Appliance Profiles




**(a)** 2014-08-01

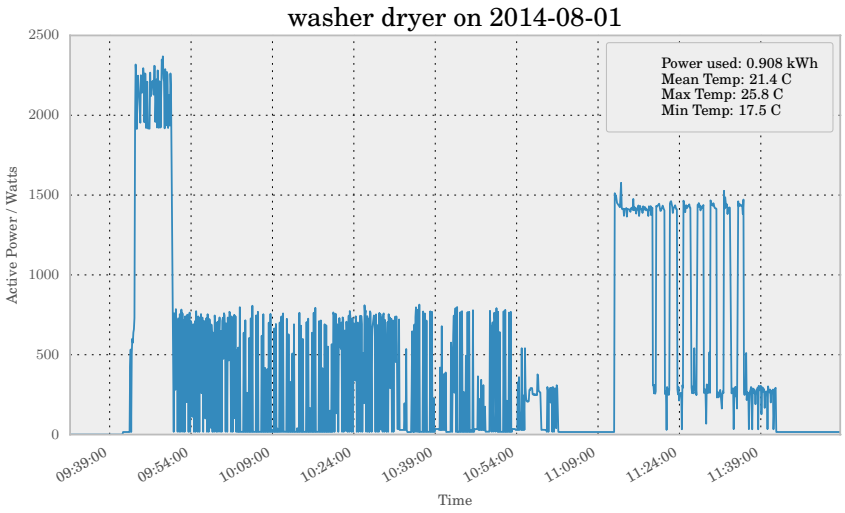



**(b)** 2014-08-13

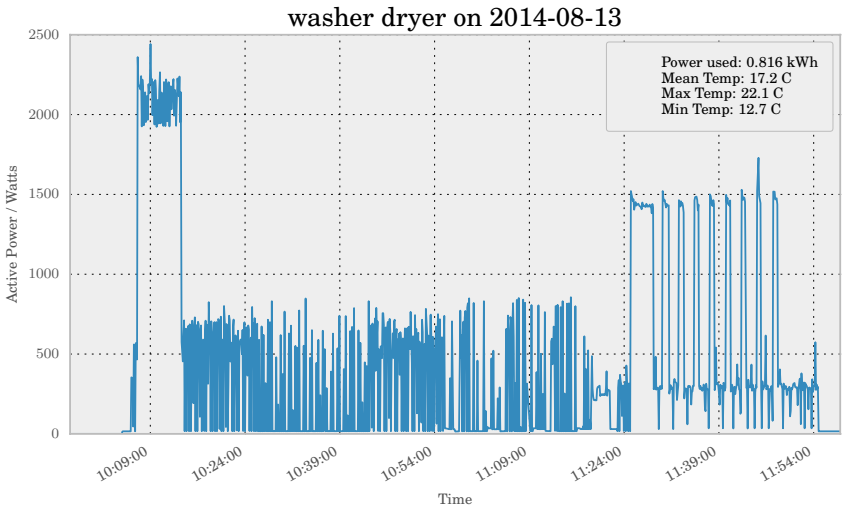



**(c)** 2014-08-19

**Figure D.1:** Appliance profiles for the dishwasher on varying dates and temperatures.




**(a)** 2014-08-01




**(b)** 2014-08-13

**Figure D.2:** Appliance profiles for the washer-dryer on varying dates and temperatures.

# Bibliography


[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 8:1027–1035, 2007.

[2] Michael Baranski and Jürgen Voss. Genetic algorithm for pattern detection in NIALM systems. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 4(2):3462–3468, 2004.

[3] Nipun Batra, Manoj Gulati, Amarjeet Singh, and Mani B Srivastava. It's Different: Insights into Home Energy Consumption in India. *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*, (August):3:1—-3:8, 2013.

[4] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. NILMTK : An Open Source Toolkit for Non-intrusive Load Monitoring Categories and Subject Descriptors. *International Conference on Future Energy Systems (ACM e-Energy)*, 2014.

[5] Mario Berges, Ethan Goldman, H. Scott Matthews, and Lucio Soibelman. Learning Systems for Electric Consumption of Buildings. In *Computing in Civil Engineering (2009)*, pages 1–10, Reston, VA, June 2009. American Society of Civil Engineers.

[6] Mario Bergés, Ethan Goldman, H. Scott Matthews, and Lucio Soibelman. Enhancing electricity audits in residential buildings with nonintrusive load monitoring. *Journal of Industrial Ecology*, 14(5):844–858, 2010.

[7] G Britain. Climate Change Act 2008 (c.27), 2008.

[8] K. Carrie Armel, Abhay Gupta, Gireesh Shrimali, and Adrian Albert. Is disaggregation the holy grail of energy efficiency? The case of electricity. *Energy Policy*, 52:213–234, 2013.

[9] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[10] P. Ciais, C. Sabine, G. Bala, L. Bopp, V. Brovkin, J. Canadell, A. Chhabra, R. DeFries, J. Galloway, M. Heimann, C. Jones, C. Le Quéré, R.B. Myneni, S. Piao, P. Thornton, Philippe Ciais France, Jan Willem, Pierre Friedlingstein, and Guy Munhoven. 2013: Carbon and Other Biogeochemical Cycles. *Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change.*, pages 465–570, 2013.

[11] A.L. Cole and A. Albicki. Data extraction for effective non-intrusive identification of residential power loads. *IMTC/98 Conference Proceedings. IEEE Instrumentation and Measurement Technology Conference. Where Instrumentation is Going (Cat. No.98CH36222)*, 2:0–3, 1998.

[12] Rob Collingridge. Raspberry Pi & Bluetooth. http://www.dreamgreenhouse.com/projects/2012/rpibt/index.php, 2012.

[13] D Comaniciu and P Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[14] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data-driven scale selection. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 438–445. IEEE Comput. Soc, 2001.

[15] Wikimedia Commons. Dbscan. https://upload.wikimedia.org/wikipedia/commons/a/af/DBSCAN-Illustration.svg, 2011. File: DBSCAN-Illustration.svg.

[16] UK Conservative Party. Invitation to Join the Government of Britain: The Conservative Manifesto 2010. pages 1–120, 2010.

[17] Martin Ester, Hans P Kriegel, Jorg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[18] Crafy Apps EU. Tasker : Android Apps on Google Play. https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm&hl=en_GB, 2014.

[19] European Parliament. Directive 2012/27/EU of the European Parliament and of the Council of 25 October 2012 on energy efficiency. *Official Journal of the European Union Directive*, (October):1–56, 2012.

[20] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, January 1975.

[21] Great Britain. Department of Energy & Climate. Smart Metering Implementation Programme Government Response to the Consultation on the second version of the Smart Metering Equipment Technical Specifications Part 2. Technical Report July, Great Britain. Department of Energy & Climate Change, 2013.

[22] Great Britain. Department of Energy & Climate Change. Energy Security Strategy. Technical Report November, Great Britain. Department of Energy & Climate Change, 2012.

[23] Great Britain. Department of Energy & Climate Change. Smart Metering Implementation Programme. Technical report, Great Britain. Department of Energy & Climate Change, 2013.

[24] Sidhant Gupta, Matthew S Reynolds, and Shwetak N Patel. ElectriSense. In *Proceedings of the 12th ACM international conference on Ubiquitous computing - Ubicomp '10*, page 139, New York, New York, USA, 2010. ACM Press.

[25] George W. Hart. Prototype Nonintrusive Appliance Load Monitor. Technical report, MIT Energy Laboratory, 1985.

[26] George W. Hart. Residential energy monitoring and computerized surveillance via utility power flows. *IEEE Technology and Society Magazine*, 8(2):12–16, 1989.

[27] George W. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[28] J a Hartigan and M a Wong. Algorithm AS 136: A K-Means Clustering Algorithm. *Applied Statistics*, 28(1):100, 1979.

[29] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[30] Iain MacLeay, Kevin Harris, Anwar Annut, , and chapter Authors. Digest of United Kingdom Energy Statistics (DUKES) 2013. Technical report, Great Britain. Department of Energy & Climate Change, 2013.

[31] ThinkTank Energy Products Incoporated. Watts up? Products : Meters. https://www.wattsupmeters.com/secure/products.php?pn=0, 2014.

[32] IPCC. 2013, Summary for Policymakers. *Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*, pages 1–29, 2013.

[33] Peter Jacob. GitHub: scikit-learn: Added new kernels to Mean Shift clustering. https://github.com/scikit-learn/scikit-learn/pull/3026/files, 2014.

[34] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2014-08-12].

[35] Jack Kelly. *Disaggregating Smart Meter Readings using Device Signatures*. PhD thesis, Imperial College London, 2011.

[36] Jack Kelly and William Knottenbelt. UK-DALE': A dataset recording UK Domestic Appliance-Level Electricity demand and whole-house demand. Technical Report March, April 2014.

[37] J Zico Kolter and Matthew J Johnson. REDD : A Public Data Set for Energy Disaggregation Research. In *SustKDD workshop*, volume xxxxx, pages 1–6, 2011.

[38] Zico Kolter, Tommi Jaakkola, and J Z Kolter. Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation. *Proceedings of the International Conference on Artificial Intelligence and Statistics*, XX:1472–1482, 2012.

[39] UK Labour Party. Labour Party Manifesto: A Future Fair For All. pages 1–78, 2010.

[40] Jian Liang, Simon K K Ng, Gail Kendall, and John W M Cheng. Load Signature Study-Part I: Basic Concept, Structure, and Methodology. *IEEE Transactions on Power Delivery*, 25(2):551–560, April 2010.

[41] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, March 1982.

[42] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[43] Nanode. What is Nanode? http://www.nanode.eu/what-is-nanode/, 2014.

[44] National Grid. UK Future Energy Scenarios 2014. *Energy*, (July):220, 2014.

[45] Robert J. Nicholls and Abiy S Kebede. Indirect impacts of coastal climate change and sea-level rise: the UK example. *Climate Policy*, 12(sup01):S28–S52, September 2012.

[46] LK Norford and SB Leeb. Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms. *Energy and Buildings*, 24(1):51–64, 1996.

[47] H. Nyquist. Thermal Agitation of Electric Charge in Conductors. *Physical Review*, 32(1):110–113, July 1928.

[48] Ofgem. Electricity Capacity Assessment Report 2014. Technical Report June, Ofgem, 2013.

[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[50] Fernando Pérez and Brian E. Granger. IPython: a system for interactive scientific computing. *Computing in Science and Engineering*, 9(3):21–29, May 2007.

[51] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, November 1987.

[52] J.S. Seabold and J. Perktold. Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference*, 2010.

[53] Nick Stallman. Gentoo Wiki Archives: TIP Bluetooth Proximity Monitor. http://www.gentoo-wiki.info/TIP_Bluetooth_Proximity_Monitor, 2008.

[54] F. Sultanem. Using appliance signatures for monitoring residential loads at meter panel level. *IEEE Transactions on Power Delivery*, 6(4):1380–1385, 1991.

[55] Warit Wichakool, Al Thaddeus Avestruz, Robert W. Cox, and Steven B. Leeb. Modeling and estimating current harmonics of variable electronic loads. *IEEE Transactions on Power Electronics*, 24(12):2803–2811, 2009.

[56] Michael Zeifman and Kurt Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE Transactions on Consumer Electronics*, 57(1):76–84, 2011.

[57] Ahmed Zoha, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. Non-intrusive load monitoring approaches for disaggregated energy sensing: a survey. *Sensors (Basel, Switzerland)*, 12(12):16838–66, January 2012.