Challenges and requirements for data-driven simulation of photovoltaic energy production and demand-side management in apartment buildings

Vebjørn Kvisli

Thesis submitted for the degree of Master in Informatics: Programming and Networks
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2019
Challenges and requirements for data-driven simulation of photovoltaic energy production and demand-side management in apartment buildings
Challenges and requirements for data-driven simulation of photovoltaic energy production and demand-side management in apartment buildings

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo
Abstract

This thesis presents challenges and necessary requirements for simulating shared photovoltaic (PV) energy production and appliance consumption scheduling in urban apartment buildings. Such simulations are useful as a tool to better understand the benefits of using PV systems that are collaboratively shared across households and how smart control systems for demand-side management can improve their utilization by reducing the amount of both excess and lack of energy at any time. An overview of factors that influence energy production and consumption in such environments is introduced and details on how these can be processed and represented in simulation software are presented. As part of such simulations, information about the energy production is necessary and this thesis proposes a method for obtaining this information. The method combines mathematical models for calculating how the sun’s angle of incidence on PV panels affect the production at any time and place, with the weather variations obtained from real measurement data from selected households in Konstanz, Germany. For the consumption part, the focus is specifically on automatic appliance load scheduling for washing machines and dishwashers based on the amount of available PV production at any time. Consumption is obtained by extracting and replicating consumption data from the same German households and classifying it into several types of washing programs. Finally, as a result of implementing the established requirements and methods presented, a simulation software called SolarSim has been developed and technical software documentations are provided for this, as well as a software for processing measurement data.
Preface

I would like to express my gratitude and appreciation to Geir Horn, my supervisor for this project. He has been a great support during the process, and has provided guidance, technical advice, and inspiration that helped drive the work in the right direction. Additionally, I would like to thank my co-supervisor Frank Eliassen for supplementary guidance.

My motivation on the topics presented in this thesis has been an important driver in the project, as they raise important questions and potential solutions to environmental issues, and the sustainability of our energy consumption in the future. Thus, I am grateful for the opportunity to make a contribution in the field.

Finally, I would like to express my appreciation to the University of Oslo for facilitating this project and providing a supportive and educational environment to work in.

– Vebjørn Kvisli
Contents

1 Introduction ................................................. 6
  1.1 Context and motivation ................................. 6
  1.2 Background ........................................... 7
  1.3 Research hypothesis and goal ........................ 8
  1.4 Approach ............................................. 8

2 Factors determining PV production and energy consumption .... 10
  2.1 PV electricity production factors ...................... 10
    2.1.1 Solar radiation .................................. 11
    2.1.2 Obstructions .................................... 11
    2.1.3 Vertical tilt angle ................................ 12
    2.1.4 Horizontal orientation angle ...................... 13
    2.1.5 Weather .......................................... 13
    2.1.6 Hardware ......................................... 15
  2.2 Energy consumption factors ............................. 15
    2.2.1 Households ...................................... 15
    2.2.2 Appliances ....................................... 16
    2.2.3 Consumption patterns ............................. 18
    2.2.4 Pricing .......................................... 19
  2.3 Factors included in SolarSim ............................ 20

3 Measurement data ........................................... 22

4 Methods for constructing PV production for simulation ........ 24
  4.1 Calculating the angle of incidence (AOI) ............... 24
  4.2 Calculating the incidence angle modifier (IAM) .......... 26
    4.2.1 IAM model ...................................... 26
    4.2.2 Alternative models ............................... 27
  4.3 Calculating the time of sunrise and sunset ............... 29
  4.4 Processing and categorizing measured production data .... 30
  4.5 Combining models and data to construct production ...... 34
  4.6 Indirect solar irradiance ............................... 37
8 Discussions and future work

8.1 Additional appliances

8.2 Machine learning for program classification

8.3 Battery storage

8.4 Indirect solar irradiance

8.5 Representative data sets

9 Conclusions
1 Introduction

1.1 Context and motivation

The world is currently facing a major challenge regarding environmental issues. Although leaders around the world have different views on the magnitude of this challenge, organizations are now initiating strategies and frameworks hoping to make a contribution, and to eventually realize a global solution. As a contribution to the Paris Agreement\(^1\) the EU has set a goal to reduce its greenhouse gas emissions by 40% of the levels from 1990, and to increase the use of renewable energy to 27% of total consumption within 2030\(^2\). Additionally, a long-term strategy has also been presented with a goal of a climate-neutral Europe by 2050\(^3\).

In order to achieve the climate goals set by the EU, as well as other organizations, it is evident that we need to increase the global awareness and priority on the use of renewable energy sources, such as solar and wind energy. Wind power systems are great for capturing clean and renewable energy, but may be impractical in urban environments due to their size and noise generation. Additionally, depending on the layout of a city and its neighborhoods, the availability of wind may be limited. Thus, photovoltaic (PV) technology is often more suited, and it is important to understand how solar energy can be utilized efficiently in these urban environments, for example on apartment buildings.

One area that is currently seeing improvement in the use of solar energy is Norway. The majority of solar energy systems in Norway have historically been small scale and often installed in vacation houses, whereas now, an increasing number of solar energy systems are being installed on large urban buildings as well \[1\]. An important contributor to facilitate efficient collaborative sharing of available energy across households may be to develop smart solutions for demand-side management, e.g. control systems that automatically schedule the consumption from individual household appliances.

---

\(^1\) The Paris Agreement - United Nations Framework Convention on Climate Change (UNFCCC): [https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement](https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement)


1.2 Background

One of the challenges with using solar energy in residential buildings is that much of the electricity is often produced when people are away, e.g. at work or in school. Storing excess electricity in batteries is a solution to this, but may be expensive and often has a limited efficiency as batteries discharge by internal leakage [2].

A good supplementary solution may therefore be to use smart appliance scheduling systems, enabling some of the consumption to be shifted to periods where the demand is lower or the supply is higher, such as during work hours. However, it should be noted that this is only suitable for ‘shiftable’ appliances, i.e. appliances that can be started at varying times without any significant inconvenience to the user, such as washing machines and dishwashers.

To be able to establish the types and significance of potential benefits of investing in PV technology and consumption scheduling systems, apartment building owners and stakeholders have a need for tools to help determine this. Software for simulation of production and consumption of energy in their particular buildings is a tool that can potentially be a predominant contributor to such decision makings. It is evident that it requires various parameters specific to both the building itself and its environment, i.e. its scenario. However, configuring scenarios is not necessarily a trivial task, and it is important to establish the requirements that must be in place and what types of parameters that should be included.

In addition to building owners and stakeholders, it may also be necessary for its residents to take part in the decision making regarding the commitment to PV system installations. This is because they most likely will have to provide a share of the financial investment. Since most residents presumably do not have any significant knowledge of the PV technology domain, simulation tools should also be provided as sufficiently simple versions with visual and intuitive interfaces, suitable for this purpose.

Data-driven simulation, which is considered in this work, is a well known method for simulating realistic situations, and is based on using real measured data to drive the simulation. However, a challenge with this method is that it is in some degree tied to the specific location and environment the data was measured in, and it needs to be processed in some way to be representative for other environments and locations.
1.3 Research hypothesis and goal

The research hypothesis for the thesis is that real energy data measured at a specific location can be used to simulate production and consumption of energy for any apartment building at any time and location, by performing suitable pre-processing and combining it with mathematical models applied to parameters specific for the environment to be simulated.

The main goal of the thesis is to explore the challenges with simulation of energy production and consumption, and to establish the requirements and parameters necessary to develop a simulation software for this purpose, and thus provide a conclusion the proposed hypothesis.

1.4 Approach

To reach the goals set for this thesis, the work process started by studying related work to get insight in the topic and learn about its background. A theoretical analysis was done to establish the various factors that influence production and consumption of PV energy. Chapter 2 gives an introduction and discusses these factors, with extra focus on some factors that are especially relevant in Norway and other high latitude locations. These factors serve as the foundation for being able to create realistic scenarios that can be used for simulation.

Furthermore, a set of methods and models that can be used to provide the production and consumption for data-driven simulation has been established. This is presented in Chapter 4 for the production data and Chapter 5 for the consumption data. More specifically, the energy production is obtained using a combination of a mathematical model for calculating the sun’s angle of incidence on PV panels, and extracting weather variations from the production of real measurement data from residential PV systems in a selection of households in Konstanz, Germany. The energy consumption for washing machines and dishwashers is obtained by extracting individual consumption profiles per wash cycle from these households, and classifying them into different washing programs based on their duration and total energy consumption.

As a result of implementing the established methods and models presented in the thesis, a simulation software called SolarSim has been developed, as well as an additional data processing software used as a tool for preparation of the data.
to be used for simulation. SolarSim is a web application that lets users configure various scenarios and simulate the production of electricity from a PV energy system, as well as the electricity consumption of a small selection of household appliances in an apartment building. It has been implemented with an existing appliance load scheduler software and aims to optimally schedule the appliances to start at various times of the day within a given period, based on the availability of solar energy throughout that day. The targeted users of SolarSim is primarily professionals in the energy industry, researchers, and people involved with the planning of apartment building construction. However, it has been developed with the goal that anyone should be able to use it, and it is characterized by user-friendly interaction and intuitive visualization elements. A technical documentation of the data processing software is found in Chapter 6 and documentation of the SolarSim application is found in Chapter 7.
2 Factors determining PV production and energy consumption

The production of electricity from photovoltaics (PV) and the consumption of energy in households is determined by several influencing factors. The focus here is on the factors that are relevant for apartment building microgrids with a locally installed PV energy system, located in an urban setting. Additionally, extra attention has been aimed at some factors that are especially relevant in Norway and other high latitude locations. The purpose of establishing these factors is to provide an overview of the various components that should be considered when developing a simulation software. Furthermore, these factors serve as the foundation for configuring scenarios, which is used as input for running the simulations.

It should be noted that some of the factors are difficult to simulate. This is among other things due to the lack of local information about specific PV systems and apartments. For example, some solar panels are obstructed by buildings and vegetation, and it may be difficult to accurately represent these obstructions in a simulation. This is especially relevant in urban environments since the density of residences are often higher. Section 2.3 gives a summary of which factors that have been implemented in the SolarSim software.

2.1 PV electricity production factors

Photovoltaic (PV) technology is one of the most viable sources of renewable energy in urban environments. This is largely due to the fact that they produce clean electricity in an efficient and environmentally beneficial manner, usually without occupying any dedicated land as they are installed on walls or rooftops [3].

Since PV modules also can be used to extract thermal energy from the solar radiation, an efficient method for making use of the generated heat should also be considered before installing a PV power system. One example of this is to flow water or air underneath the modules using photovoltaic thermal collectors [4]. However, this thesis will only focus on the electricity production.

How much electricity a PV power system will generate depends on several factors, including solar radiation, obstructions, vertical tilt angle, horizontal orientation angle, weather, and the hardware used.
2.1.1 Solar radiation

It is clear that the amount of available solar radiation is the most important factor determining the electrical output of a PV power system [3]. Depending on the operating climate and the type of solar cells used, a typical PV module only transforms 6-20% of the solar radiation received into electricity, and the rest is transformed into heat [4].

The amount of solar radiation received depends on time and place and often varies significantly over a year. In Norway, the availability of solar radiation is a challenge because of the way it is distributed throughout the year. Although the yearly amount of solar radiation in Norway is similar to that in Central Europe, it is substantial in the summer and limited in the winter due to the high latitude [1]. Because of this, in addition to the differences in heating needs, it is reasonable to expect that buildings with a PV power system installed in Norway will rely more on the external power grid in the winter than in the summer.

In terms of solar radiation stability, the predominant influencer is the presence of clouds blocking the sunlight, producing an arbitrary production profile. The total irradiance received by photovoltaic surfaces is the sum of direct solar radiation and indirect diffuse solar radiation. The ratio of diffuse radiation is up to 100% of the total irradiance on completely overcast days and there is always at least 10% diffuse radiation, even on days with clear skies [5]. This is because some of the radiation is also absorbed or reflected by atmospheric variations. The amount absorbed or reflected is primarily determined by the mass of air, amount of water vapor, aerosols, and ozone [6]. The impact these variations have on PV performance is generally small, but may introduce certain production patterns based on the geographic location and seasons.

2.1.2 Obstructions

As discussed, solar radiation is the most important factor for the production of electricity in PV power systems. A concern related to this is obstructions of the solar radiation. Neighboring buildings can form obstructions to the sun, diminishing the solar irradiance on the PV panels [3]. Additional obstructions may be caused by trees and other vegetation, but these are less likely to be an issue in the context of apartment buildings of a significant height. Thus, whether PV panels should be
installed on a building’s rooftop or walls, naturally depends on the layout of the building and its neighboring buildings and vegetation.

To help determine the level of obstruction for a PV power system an ‘obstruction angle’ (OA) can be used. Yun and Steemers defined an OA as “the altitude angle subtended from the mid-point of a window to the mean roof height of the adjacent buildings” [3]. Furthermore, the loss of electricity generation from PV panels due to the OA is intensified as latitude rises. This means that the OA is a more significant factor in Norway than other lower-latitude locations.

For instance, in Oslo (59° N) the output of a PV panel was shown to be reduced by 82% when going from an OA of 0° to 61°, while in Milan (45° N) it was reduced by 69%, and OA thresholds for installing urban PVs in Oslo and Milan was suggested to be maximum 18° and 38° respectively [3].

2.1.3 Vertical tilt angle

The vertical tilt angle of PV panels is important to consider in order to fully utilize the available solar radiation. The optimal tilt angle is dependent on the geographic location, and availability of sunlight in that location. As discussed, in Norway there is a large difference in available sunlight from summer to winter time, because of the changing height of the sun. More specifically, in Oslo the sun height at noon is at around 55° in the summer and only around 10° in the winter [1]. Because of this change it is clear that the optimal tilt angle for PV panels varies, and the maximum electrical output would be achieved by continuously adjusting the tilt angle to the optimal position throughout the year. Researchers have reported that this approach might seem unpractical, but that making adjustments per month or season can give a significant increase in electrical output at a low investment [7].

In a study performed in Southern Norway (58° N) a range of different tilt angles were examined for using PV panels on a grid-connected residential house [8]. The study suggested that the optimal fixed tilt angle for maximum annual output in this area is 39°, where 0° is horizontal and 90° is vertical. However, looking at each season of the year, it was indicated that the optimal tilt angle is around 30° in the summer and close to 90° in the winter. In the early spring and late autumn, around 60° appeared to be optimal.

Another study performed in Sanliurfa, Turkey (37° N) determined that the
monthly optimal tilt angle ranged from 13° in June to 61° in December [7]. Comparing the studies in Norway and Turkey, a significant difference in optimal tilt angles are observed because of the difference in latitude. Thus, determining one optimal tilt angle for Norway might not be practical, as significant differences are expected between northern and southern parts of the country. However, the suggested optimal tilt angles in Southern Norway give a good indication of how much it varies throughout the year.

2.1.4 Horizontal orientation angle

When designing the layout of a PV power system, the horizontal orientation angle (azimuth) of the PV modules is also a principle factor, as it regulates the amount of solar radiation received by the modules [3]. Therefore, a PV panel should be installed in the most optimal orientation to increase the electrical output, which varies depending on geographic location.

In general, the optimal orientation is facing Equator, i.e. due north in the Southern Hemisphere and due south in the Northern Hemisphere [5]. However, Yun and Steemers suggested that PV panels in the Northern Hemisphere can produce at least 80% of their maximum electrical output if they are installed with a horizontal orientation within 120° and 240°, where 180° represents due south [3]. In other words, it might be acceptable to install PV panels within a 60° angular distance from Equator. Using wider angular distances than 60° will result in significant reductions in electrical output. Naturally, the decision also depends on whether or not obstructions are present.

2.1.5 Weather

Weather is an important factor for the performance of PV systems because it affects the amount of solar radiation received by the PV panels. Numbers from Statistics Norway shows that there were on average 219, 196 and 215 days with precipitation in Norway in 2012\textsuperscript{4}, 2014\textsuperscript{5}, and 2016\textsuperscript{6} respectively, based on 12 different location. This relatively high number indicates two things. Firstly, there is a need for

\begin{itemize}
  \item Statistics Norway 2012: \url{www.ssb.no/a/aarbok/tab/tab-026.html}
  \item Statistics Norway 2014: \url{www.ssb.no/233220/nedbor-sa-26}
  \item Statistics Norway 2016: \url{www.ssb.no/314450/nedbor-sa-26}
\end{itemize}
electricity storage, where electricity can be saved on sunny days and consumed on cloudy or rainy days. Secondly, detailed weather forecasts are necessary to achieve accurate predictions of the electrical output of PV systems.

As one would expect, on sunny days the power output of a PV system roughly correlates with the solar radiation intensity, but on cloudy days the solar radiation is partially blocked by the clouds and the output drops drastically [9]. On rainy days the output drops even further.

Another weather-related factor worth mentioning is snow coverage. In Norway there is typically a significant, though varying, number of snowfalls during the winter season, which can cover the surface of PV panels. As an example, an installation of PV modules in Kristiansand (58° N) reported to experience a period of 13 consecutive days in March with near minimal production after a recent snowfall covered the modules [10]. Considering that Kristiansand experiences relatively mild winters regarding the number of snowfalls compared to other parts of Norway [10], one would expect similar or longer periods of snow coverage other places in Norway. Naturally, a solution to this problem is to manually clean the modules, removing the snow. However, this is not always a practical solution, especially for tall apartment buildings.

Temperature is also a factor for the electricity production of a PV power system. The performance of solar cells decreases as the operating temperature of the cells increases [4]. Because of this, on the Northern Hemisphere, the performance generally increases with increasing latitude and altitude, because of the differences in temperature. Thus, the relatively cold temperatures in Norway is well suited for PV installations as it contributes to cooling down the solar cells. In addition to latitude and altitude, the operating temperature of a PV module also correlate with other variables such as wind speed, amount of solar radiation, and material properties [4].

In order to use weather as an input variable for simulations, it needs to be represented as defined non-ambiguous values and a definition of these is necessary. This is not a trivial task considering the complexity of the factors that determine the weather. One example of how the weather can be represented is to use a percentage of solar radiation blocked due to the weather. Another example is to define a set of weather types. In this work, the latter has been used and the weather types have been defined as sunny, partially cloudy, cloudy, and rainy. The
methods for defining and assigning weather types is further described in Section 4.4.

2.1.6 Hardware

PV modules are manufactured with a range of different technologies, and this may affect their performance in various climates. For example, it was reported that a PV array of triple junction amorphous silicon (a-Si) modules performed up to 20% better than one with poly-crystalline silicon (p-Si) modules in the tropical climate of northern Australia [11].

Because of this it may be useful to estimate the electricity output of a specific type of PV module under realistic operating conditions. Manufacturers label PV modules with a rated power indicating its efficiency under standard test conditions (STC) [11]: an irradiance of 1000 Wm$^2$, an air mass of 1.5 (AM1.5) and a cell temperature of 25°C. Testing under these conditions ensures that PV modules can be evaluated independently of external factors and provides a good indication of their performance. However, they do not always correspond with realistic operating conditions.

2.2 Energy consumption factors

How much electricity an apartment building consumes will typically vary as a result of the unpredictable nature of human behavior, which determines how and when we use different electrical appliances. However, several factors can be established, which would give a good indication of the consumption level. Some of the factors are constant, such as the number of households in the apartment building and the floor area of each household.

2.2.1 Households

Information about the households in an apartment building is important in order to estimate the total electricity consumption. A survey by Statistics Norway\footnote{Statistics Norway - “Energy consumption in households, 2012”: \url{https://www.ssb.no/en/energi-og-industri/statistikker/husenergi/hvert-3-aar/2014-07-14}} showed a downward trend in energy consumption in Norwegian households over
the last couple of decades. The reason for this is presumably a milder climate and more energy aware consumers taking steps to reduce their consumption. The survey also provides statistics of the average energy consumption per square meter as shown in Table 1. These values can potentially be used to get a reasonable estimate of the energy consumption for any sized household. However, the data is specific to Norwegian households, and may not necessarily coincide with the energy consumption in other countries due differences in climate, economy, culture, etc.

The survey also showed that apartments consume most of the energy as electricity, and they consume significantly less energy than farm houses, detached houses and row houses overall, as shown in Figure 1. The reasons for this are mainly that compared to these household types, apartments usually have a smaller dwelling area, fewer people living there, and less exterior walls exposed to heat loss. These differences are important to take into consideration as apartments are the household type of interest in this thesis.

Household consumption is also significantly influenced by the number of people living there. For example, the average energy consumption per square meter in Norwegian households of any type in 2012 ranged from 172 kWh for households with one resident to 207 kWh for households with five or more residents.

### Table 1: Average energy consumption (kWh) by energy product, per m² dwelling area, for all Norwegian house types (Source: Statistics Norway)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>All energy commodities</td>
<td>207</td>
<td>212</td>
<td>211</td>
<td>203</td>
<td>186</td>
<td>190</td>
<td>181</td>
<td>185</td>
</tr>
<tr>
<td>Electricity</td>
<td>167</td>
<td>169</td>
<td>169</td>
<td>164</td>
<td>145</td>
<td>147</td>
<td>146</td>
<td>149</td>
</tr>
<tr>
<td>Oil and kerosene</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>11</td>
<td>13</td>
<td>12</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>Wood, coal and coke</td>
<td>25</td>
<td>27</td>
<td>26</td>
<td>28</td>
<td>27</td>
<td>30</td>
<td>27</td>
<td>27</td>
</tr>
</tbody>
</table>

2.2.2 Appliances

How many and what types of electrical appliances a household has is important to establish in order to determine its consumption. This is naturally because it is the appliances themselves that consumes electricity. Halvorsen showed in 2012

---

that as a result of increasing household incomes and decreasing appliance costs, there has been a large increase in the percentage of households that own certain appliances [12]. For example, the percentage of Norwegian households that owned a washing machine gradually increased from 0% to approximately 93% from 1950 to 1980. Other basic appliances such as refrigerators, TVs, stoves and freezers has seen similar increases.

Furthermore, Halvorsen pointed out that the energy efficiency is generally higher in newer household appliances, and that this increase in efficiency roughly correlates with the increase in usage of newer appliances such as PCs and mobile devices. These two opposing forces resulted in a relatively stable electricity consumption level in Norwegian households from approximately 1986 to 2008, after a period a steady increase from 1960 to 1986.

The share of electric vehicles (EV) also appear to be increasing, as part of the shift towards more environmentally friendly transportation. Although it is difficult to predict the exact speed of EV’s increase in the global share of personal and commercial transportation in the future, it is safe to assume that it will become significantly more popular in the next few decades. As a result of this, the power grids may experience higher demand, as EVs often require frequent charging. A particular challenge related to this may be that EV owners prefer to charge their cars at specific times of the day, e.g. when they come home from work in the afternoon.
2.2.3 Consumption patterns

The energy consumption in Norwegian households varies in certain patterns during a given time period, based on various factors. The predominant factor is outdoor temperature. Halvorsen reported that there is a strong negative correlation (-0.97) between average outdoor temperature and average consumption throughout a year because of the need for heating (based on measurements done in 2006) [12]. In general, this means that when the temperature is lower in the winter, the consumption is higher and when the temperature is higher in the summer, the consumption is lower. However, it was also estimated that above a certain temperature (around 20°C) the average consumption starts increasing, as a result of cooling needs [12]. Because heating is the predominant portion of the total energy consumption, outdoor temperature is an important factor to consider.

Furthermore, it is also relevant to look at the consumption over a shorter time period, such as a day. For residential households, Halvorsen showed that on an average weekday there is a peak in consumption in the afternoon and evening, and a slightly lower peak in the morning [12]. The consumption on weekends are mostly similar, but the morning peak seems to occur a couple of hours later.

In Norway, Advanced Metering Systems (AMS)⁹ are being installed in all households within 1 January 2019 in the form of ‘smart meters’. This provides better information of the user’s consumption and better opportunities for the consumers to participate in demand response. The data generated from these smart meters are useful for monitoring a household’s total electricity consumption. However, when it comes to scheduling the consumption of individual appliances, detailed information at appliance level is needed, and measurement of the consumption of each individual appliance is necessary.

The load profiles of individual appliances in a household defines the temporal characteristics of its consumption [13]. These load profiles can potentially reveal additional patterns in energy consumption useful for predicting and scheduling future consumption, such as how much, when, and how often people tend to use certain appliances. In this thesis, load profiles have been extracted from extensive measurements of appliance energy consumption, and used to provide the energy

consumption for simulations of apartment buildings, further described in Chapter 5.

It should be noted that measured consumption data and its patterns also depend on factors such as people’s personal preferences, culture, and geographical location. Thus, data from several parts of the world are needed to give a good representation of these factors.

### 2.2.4 Pricing

In Norway, the electricity price that a typical consumer has to pay is divided into three parts: electricity price, taxes, and grid rent. The overall average price of electricity for Norwegian households, including taxes and grid rent, was 0.965 NOK per kWh in 2017, of which 0.342 NOK was the actual electricity price\(^ {10} \). However, the price depends on the type of contract a customer has. In contrast to some countries, all three parts of the Norwegian electricity price is determined by the amount of electricity used. Presumably, using more electricity from locally installed PV systems would reduce the need for externally generated electricity accordingly, and this would not only reduce the cost of the electricity price, but also taxes and grid rent.

It is reasonable to assume that an apartment building microgrid with a PV power system installed, still will have to be connected to the power grid. Even if a building can achieve self sustainability for substantial periods, there will always be an uncertainty regarding the weather. In the case of a long period with very little solar radiation, and limited battery storage, the building will have to rely on electricity from the power grid. In order to make the best use of available PV generated electricity, scheduling the consumption is necessary. However, this requires a certain degree of flexibility and cooperation from the end consumers. Thus, we need to understand what types of incentives that are best suited for encouraging this.

Flexible pricing is one incentive that can be used to motivate consumers to schedule their consumption more efficiently. In general, a low price can be offered to the consumers when the demand is low or the PV generation is high. Corre-

spondingly, a higher price can be offered when demand is high or PV generation is low. This approach may help to cut the load peaks in the power grid, by reducing people’s need for externally generated electricity. However, in terms of altering electricity consumption, household consumers respond relatively weakly to changes in price. Halvorsen showed that when the price increases by 1%, the consumption in the same hour is reduced by 0.06% [12]. Additional incentives may therefore be necessary to reduce the consumption further.

2.3 Factors included in SolarSim

As discussed in Sections 2.1 and 2.2, there is a large number of factors determining PV production and household energy consumption. For the development of the SolarSim application, most of the significant factors has been included. However, several of the factors are for this application too complex to accurately model, such as people’s personal preferences, their response to price changes, the ratio of diffuse radiation on solar panels, and the level of obstruction for a PV system. To conclude and summarize the chapter, the following parameters have been included in SolarSim to represent the factors, and are provided as inputs for the users. For details of the individual parameters, see the SolarSim documentation in Chapter 7.

- **Scenario:**
  - PV system’s rated output
  - Vertical tilt angle
  - Horizontal orientation angle
  - Weather type
  - Date
  - Location

- **Apartments:**
  - Number of apartments
  - Number of residents
- Apartment size
- A list of appliance runs

**Appliance runs:**

- Type of appliance
- Program
- Earliest start time
- Latest time of completion
3 Measurement data

The data set used in this thesis contains household appliances’ energy consumption and photovoltaic energy production in various time resolutions, and is a result of the project Collaborating Smart Solar-powered Microgrids (CoSSMic)\textsuperscript{11}. This project aims to maximize the self-consumption of solar power in entire neighborhoods, rather than individual households, by coordinating their energy use and storage [14]. There is a total of 11 households in Konstanz, Germany included in the data set, of which six are residential. Since industrial appliances and PV systems may have different properties, which are not necessarily representative for residential households, only these six have been used in this thesis. The data set and its documentation is available online\textsuperscript{12}.

The structure of the data is in the form of cumulative time-series files with regular time intervals and a time stamp for each sample. The data quality is not perfect, due to varying starting points of the time-series for different households, and the presence of data gaps filled with linear interpolation. However, since the data is cumulative, the overall energy consumption and production is retained. The data values are the actually transmitted energy values in kWh, measured by MID-certified\textsuperscript{13} metering devices.

These time-series files have been analyzed, processed, and categorized with a data processing software that has been developed specifically for this work, in order to be used for data-driven simulation. It should be noted that these procedures cause the production data used for simulation to be a modified version of the data, and is not a direct representation of the original. It is modified to simulate the specific scenario parameters given by the user. The simulated consumption, however, is only extracted and categorized, not directly modified.

The data set contains measurements of a range of devices and appliances. In this thesis the focus is on washing machines, dishwashers, and PV systems. Table 2 shows an overview of all the individual appliances and PV systems used in this work, as well as which time resolution was used for sampling. Generally, the highest time resolution (1 minute) holds the most information and is desired, but for the PV systems, the 3 minute resolution was chosen because it appeared to

\textsuperscript{11}CoSSMic project: https://cossmic.eu
\textsuperscript{12}CoSSMic household data set: https://doi.org/10.25832/household_data/2017-11-10
\textsuperscript{13}Certified by the Measuring Instruments Directive (MID)
<table>
<thead>
<tr>
<th>Device</th>
<th>Name (abbr.)</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Washing machines</td>
<td>DE_KN_residential1_washing_machine (WM1)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential2_washing_machine (WM2)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential3_washing_machine (WM3)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential4_washing_machine (WM4)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential5_washing_machine (WM5)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential6_washing_machine (WM6)</td>
<td>1 minute</td>
</tr>
<tr>
<td>Dishwashers</td>
<td>DE_KN_residential1_dishwasher (DW1)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential2_dishwasher (DW2)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential3_dishwasher (DW3)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential4_dishwasher (DW4)</td>
<td>1 minute</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential5_dishwasher (DW5)</td>
<td>1 minute</td>
</tr>
<tr>
<td>Photovoltaics</td>
<td>DE_KN_residential1_pv (PV1) ( \beta = 30^\circ, \gamma = 175^\circ )</td>
<td>3 minutes</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential3_pv (PV3) ( \beta = 5^\circ, \gamma = 20^\circ )</td>
<td>3 minutes</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential4_pv (PV4) ( \beta = 28^\circ, \gamma = 208^\circ )</td>
<td>3 minutes</td>
</tr>
<tr>
<td></td>
<td>DE_KN_residential6_pv (PV6) ( \beta = 40^\circ, \gamma = 210^\circ )</td>
<td>3 minutes</td>
</tr>
</tbody>
</table>

Table 2: The appliances and PV systems used in this work. The installation of the PV systems are specified with their vertical tilt angle \( \beta \) and horizontal (azimuth) orientation angle \( \gamma \).

have the least amount of data gaps. Note that out of the six residential households, four had a PV system installed, and one household’s dishwasher is excluded due to unexpected, and possibly unreliable consumption values. In general, the data with a resolution of 3 minutes was measured from April 15, 2015 to February 9, 2017. Data with a resolution of 1 minute was initiated a little later and was measured from October 26, 2015 to February 9, 2017. However, the start and end date for individual devices vary within this period.
4 Methods for constructing PV production for simulation

In order to construct the production of electricity from PV systems that can be used for simulation, it is necessary to generate a realistic production profile with information about how much is produced throughout the simulation period. Solar panels’ efficiency are dependent on the factors discussed in Section 2.1, some of which can be difficult to replicate accurately. For example, because of the arbitrary nature of weather variations, clouds blocking the solar radiation cause irregular production. Although a perfectly accurate prediction of a PV system’s output is difficult to perform, a realistic approximation in simulations is possible.

In general, the approach used to construct production profiles for simulation is a combination of mathematical models and the use of measurement data. First, the impact of the PV system’s tilt and orientation angles is evaluated. These angles determine the sun’s angle of incidence on the solar panels. Then, a theoretically optimal production profile is calculated for the specific scenario parameters to simulate, based on the angles of incidence throughout the simulation period. Furthermore, fluctuations obtained from the real measurement data are applied to these optimal values to replicate varying weather conditions. These fluctuations are retrieved from pre-categorized production data, and they are scaled to match the simulation scenario parameters.

A method for theoretically calculating the times of sunrise and sunset at the location to simulate is also provided in order to make sure there is only production when the sun is up. These times vary significantly throughout the year, especially at latitudes far from Equator. At extreme latitudes, there are in certain periods no sunrise or sunset, due to Earth’s axial tilt.

4.1 Calculating the angle of incidence (AOI)

The tilt and orientation angles of PV modules affect the angle at which it receives solar radiation. Generally, the performance is best when the sun is shining directly onto the panels, i.e. when the panels are perpendicular to the solar beam. However, solar panels with fixed tilt and orientation angles are rarely perpendicular to the solar beam as the sun continuously moves across the sky. Thus, it is useful to
calculate the angle of incidence (AOI) so that a performance factor can be obtained as a function of the angle.

The AOI is defined as the angle between the solar beam and the panels’ normal vector. For $AOI = 0^\circ$ the panels are perpendicular to the solar beam, and for $AOI > 90^\circ$ the panels are facing away from the sun and is not receiving any solar radiation. De Soto et al. showed that because of the reflection of solar radiation on PV modules, angles greater than approximately $65^\circ$ have a significant effect on the amount of solar radiation transmitted through the cover of the PV module, and angles less than $65^\circ$ have little effect [15].

Calculating the angle of incidence of PV modules is not only dependent on their tilt and orientation angles, but also the time and location. Twidell and Weir derived Equations (1-3) for calculating the angle of incidence ($\theta$) when the tilt, orientation, time, and location of a PV module is known [5, pp. 42-48]. Eq. (1) takes into consideration Earth’s declination ($\delta$) at the given time of the year and the solar hour angle ($\omega$) at the given time of day and location. Latitude, tilt, and orientation are represented by $\phi$, $\beta$, and $\gamma$ respectively.

$$\cos \theta = (A - B) \sin \delta + [C \sin \omega + (D + E) \cos \omega] \cos \delta \quad (1)$$

where

\[
A = \sin \phi \cos \beta \\
B = \cos \phi \sin \beta \cos \gamma \\
D = \cos \phi \cos \beta \\
E = \sin \phi \sin \beta \cos \gamma \\
C = \sin \beta \sin \gamma
\]

Furthermore, Twidell and Weir showed that Earth’s declination can be calculated using Eq. (2), where the angle between Earth’s equator and Earth’s plane of revolution around the sun is $\delta_0 = 23.45^\circ$ and $n$ is the day of the year ($n = 1$ on 1 January). The resulting declination $\delta$ represents the angle between the sun’s radiation and Earth’s equatorial plane, and it varies smoothly throughout the year, from $23.45^\circ$ on 21 June to $-23.45^\circ$ on 21 December. On 21 September and 21 March $\delta = 0$.

$$\delta = \delta_0 \sin[360^\circ(284 + n)/365] \quad (2)$$
Finally, the solar hour angle $\omega$ represents the angular distance that Earth has rotated since solar noon, and can be calculated by using Eq. (3), where $t_{solar}$ is the local solar time in hours at the location of interest. The term $15^\circ/h$ comes from the fact that Earth rotates $15^\circ$ every hour ($360^\circ$ every 24 hours).

$$\omega = (15^\circ/h)(t_{solar} - 12h)$$  \hspace{1cm} (3)

### 4.2 Calculating the incidence angle modifier (IAM)

Once the angle of incidence has been calculated, an expression is needed to convert the angle into a performance factor for the PV panel: an incidence angle modifier (IAM). Several different expressions with various degrees of complexity can be used for this purpose. Some of the expressions that have been proposed use measured specifications of the PV panel as parameters, while other work by simply fitting a few parameters. Although they produce fairly similar results, they vary in complexity and accuracy, and which model to use depends on the application.

#### 4.2.1 IAM model

For the calculation of the incidence angle modifier, a model proposed by De Soto et al. is used in this thesis [15]. This model has a sufficient accuracy for all AOI values, and a sufficient, but manageable complexity, and is based on Snell’s law and Bougher’s law. In addition to the angle of incidence, the model needs three parameters to determine the IAM: an index of refraction for the cell surface, a glazing extinction coefficient, and a glazing thickness.

The incidence angle modifier is calculated using Equations (4-7). Note that these include a couple of corrections to errors present in [15], proposed by PVPMC\(^{14}\). Specifically, in Eq. (7) $n$ is substituted by $\frac{1}{n}$, and Eq. (6) is an alternative to calculating $\tau(0)$, instead of using Eq. (5) for $\theta = 0$ (which gives division by zero).

The incidence angle modifier ($K_{\tau_{aoi}}$) is given by Eq. (4) as the ratio between the surface’ transmittance at the angle of incidence $\tau(\theta)$ and its transmittance when normal to the sun $\tau(0)$, where $\theta$ is the angle of incidence.

\[ K_{\tau_\alpha}(\theta) = \frac{\tau(\theta)}{\tau(0)} \quad (4) \]

An approximation of \( \tau(\theta) \) considering both reflective and absorption losses is found using Eq. (5)

\[
\tau(\theta) = e^{-(KL/\cos \theta_r)} \left[ 1 - \frac{1}{2} \left( \frac{\sin^2(\theta_r - \theta)}{\sin^2(\theta_r + \theta)} + \frac{\tan^2(\theta_r - \theta)}{\tan^2(\theta_r + \theta)} \right) \right] \quad (5)
\]

where \( K \) is the glazing extinction coefficient, \( L \) is the glazing thickness, and \( \theta_r \) is the angle of refraction. Furthermore, the transmittance when normal to the sun \( \tau(0) \) is given by Eq. (6) as \( \theta \) approaches 0.

\[
\tau(0) = \lim_{\theta \to 0} \tau(\theta) = e^{-KL} \left[ 1 - \left( \frac{1 - n}{1 + n} \right)^2 \right] \quad (6)
\]

Finally, Eq. (7) is used to determine the angle of refraction \( \theta_r \) from Snell’s law.

\[
\theta_r = \arcsin \left( \frac{1}{n} \sin \theta \right) \quad (7)
\]

where \( n \) is the refraction index of the cell surface.

The values used to calculate incidence angle modifiers in this thesis were suggested as typical input parameters in [15]. These values are \( n = 1.526 \) for the refraction index of glass, \( K = 4 m^{-1} \) for the glazing extinction of ‘water white’ glass, and \( L = 2 \) mm for a glazing thickness reasonable for most PV cell panels. Figure 2 shows the model given by these equations and parameter values for AOI values between 0 and 90 degrees.

### 4.2.2 Alternative models

As several different models for calculating the incidence angle modifier exist, other models may produced slightly different results. An alternative IAM model for estimating the performance factor was developed by Souka and Safwat in 1966 [16]. This approach was later adopted by ASHRAE\(^{15} \) and is now known as the ASHRAE incidence modifier [17]. It is a simple function given by Equation (8),

\(^{15}\)American Society of Heating, Refrigeration and Air Conditioning: www.ashrae.org
where $\theta$ is the angle of incidence, and it works by fitting one parameter $b_0$. However, it is not suited to be used when the angle of incidence is close to $90^\circ$, as this yields negative IAM values. Figure 3 shows this model for AOI values between 0 and 90 degrees using several examples of $b_0$ values.

Another example of an alternative model is the mathematical IAM model proposed by Martin and Ruiz [17] which takes into consideration the measured specifications of the PV panel, such as the weighted reflectance, and the air-to-solar cells transmittance and absorptance of the PV module. It was obtained from optical analysis of commercial PV modules of various technologies, and avoids the problems mentioned for the ASHRAE incidence modifier. It is given by Equation (9), where $\theta$ is the angle of incidence and $a_r$ is a coefficient that must be fit for each use case. Figure 4 shows this model for AOI values between 0 and 90 degrees using several examples of $a_r$ values.
Figure 3: IAM model by Souka and Safwat [16], known as the ASHRAE incidence modifier for AOI values between 0 and 90 degrees, and several examples of $b_0$ values

\[
IAM = \frac{1 - \exp \left(-\frac{\cos(\theta) / a_r}{b_0}\right)}{1 - \exp \left(-\frac{1}{a_r}\right)}
\] (9)

4.3 Calculating the time of sunrise and sunset

Considering the framework for calculating the angle of incidence described in Section 4.1 is in place, calculating the time of sunrise and sunset at a given location and time of the year is a relatively simple task. This can be done by continuously calculating the angle of incidence of a fictional solar panel that lies flat on the ground throughout a day (tilt $\beta = 0$). Sunrise is at the time when the resulting angles of incidence transition from greater than 90° to less than 90°. Similarly, sunset is at the time when the angles transition from less than 90° to greater than 90°. This works because the normal vector of a solar panel with no tilt is pointing straight up and is perpendicular to the horizon. Thus, when the angle of incidence is 90° we know that the sun is at the horizon.

It should be noted that this approach only applies to the idealized, horizontal horizon. The visible horizon is often a result of vegetation, buildings, and mountains blocking the true horizon. As a result of this, the sun may in practice rise later and set earlier than expected. Additionally, the sun is treated as being either up or down, while in reality there is a period of a few minutes where the sun is
Figure 4: IAM model by Martin and Ruiz [17] for AOI values between 0 and 90 degrees, and several examples of $a_r$ values

only partially visible.

4.4 Processing and categorizing measured production data

Since the variation of solar irradiation on PV modules due to local weather fluctuations are unpredictable, it is difficult to reproduce theoretically. Although meteorologists can predict the weather in large areas, such as for a city, this may differ from the local weather variations experienced by individual PV systems. For example, on a partially cloudy day the electricity production can change between a high and low electricity output every time a small cloud blocks the modules’ view to the sun.

Therefore, the real measurement data of PV production has been used here to emulate these fluctuations, and this results in more realistic simulations. However, it should be noted that the characteristics of the data is specific to the location it was measured (Konstanz, Germany), and may not perfectly represent the climate of other locations.

As described in Chapter 3, the data is provided as cumulative time-series with time stamps for each sample. Figure 5 shows the time-series for the PV system PV1. This shows a clear example of the yearly and daily variation in PV production, and it can be observed that significantly less energy was produced in the
winter months. This is as expected, since Konstanz has a latitude of about 47°. Presumably, locations with higher latitude would see a similar pattern, but with even more variation, locations around the equator would see little variation, and locations with negative latitudes would see variation with the opposite pattern.

Figure 5: Example of photovoltaic energy production time-series. The left figure shows the yearly variation and the right figure shows an excerpt of approximately 17 days where the daily variation can be observed (PV system: PV1).

The first step in processing the data was to split the time-series into 24 hour production profiles, starting at midnight. This has been done by iterating over all samples and saving a new profile when the time stamp for the local time indicated that it was midnight. Additionally, during this iteration the profiles were categorized into four seasons by checking the month number of the time stamps. Each season is defined as having three month, where winter corresponds to December, January, and February. Figure 6 shows an example of this for the data from PV1, where clear differences can be observed between the seasons.

The reason for separating profiles by season is to effectively eliminate it as a factor of the production data. This is desired because the most important effects of seasonal changes are already considered in the calculation of AOI for the location to simulate, and the only factor of interest in the measured data is the variation due to the weather. Thus, production profiles can be compared separately within each season when normalizing the values, which is the next step in the data processing. It should be noted that there is still variation across profiles within each season, which is due to differences in the time of year, not the weather. As an example of this, the first day of spring has a shorter period with sunlight compared to
the last day of spring. One solution to this may be to divide the profiles into smaller groups, e.g. one for each month of the year. However, due to a limited amount of measured production data, this may result in too few profiles in each month, which in turn may generate too little variation when performing multiple consecutive simulations for one specific month.

Next, the production profiles has been normalized and given values between zero and one. This was done separately for each of the four PV systems, and separately for each season. Before normalizing, the profiles were converted from cumulative to non-cumulative values for each sample by calculating the difference between two adjacent values. For each PV system and season, the normalization was done by iterating over each sample from all profiles simultaneously and selecting the highest value in each sample. The selected values resulted in a new profile which represented that PV system and season’s optimal production for one day. This is illustrated with an example in Figure 7, where the highest production values for PV1’s autumn profiles are represented by the red line. All profiles were then normalized relative to this, meaning that their values were divided by the cor-
responding values in the optimal profile. The result was normalized values between zero and one that represent how much energy the module produced, relative to its maximum measured production at the same time of day. This approach worked well for this purpose since it ignores local obstructions such as mountains by only caring about the relationships between the values of the production profiles and the maximum measured production values specific for that particular PV system.

There are two reasons why normalization has been done. The first reason is to remove the information about how much energy the PV systems actually produced. This is not useful information in this particular application of the production profiles, as it is only the variation that is of interest. The normalized values can then be scaled up to match any combination of PV parameters set for a simulation. The second reason is to eliminate any differences in surroundings, the installation, the type of solar panels, etc. between the four PV systems.

Finally, since a weather type must be specified for the simulation, the profiles have been categorized into a selection of different weather types. The choice of weather types was not obvious, since the transitions between weather types are in reality often vague or ambiguous. In the SolarSim application, four weather types have been defined, ranging from highest to lowest production: sunny, partially cloudy, cloudy, and rainy. Although more complex weather types may occur in reality, these are sufficient for simulation purposes because they are common terms and presumably easy to understand for all users.

The process of how the weather types was assigned is fairly simple. The normalized production profiles were first turned into cumulative profiles by accumulating all the previous values together. Then, the difference between the highest and the lowest accumulated value of the profiles was calculated and this difference is divided into four equal segments, each representing one weather type. The profiles were then assigned the weather type that corresponds with their accumulated values. Figure 8 shows an example of this by displaying PV3’s summer profiles with
accumulated normalized values, and a unique color for each weather type.

It should be noted that since this categorization only uses the total accumulated values of the normalized profiles to determine their weather type, periods with lower or higher production than expected for the given weather type can occur. However, this allows for realistic simulations, as the days we consider to be sunny or clear may have periods when clouds block the solar radiation, and the days we consider to be cloudy or overcast may have periods where the solar radiation travels through small gaps in the cloud cover.

4.5 Combining models and data to construct production

To construct the energy production to be used in simulations, the mathematical models for AOI and IAM is combined with the processed measurement data, and this should be done dynamically every time a user starts a simulation, rather than as preparation in advance. This is because it is dependent on the specific scenario.
parameters chosen by the user. The following description explains in detail the process used to combine the models and data, and thus creating the production profile that can be used for simulation. Note that the process is applied to the non-cumulative versions of the processed measurement data, not the accumulated values.

1. Pick a random production profile (selected profile $sp$) with normalized values from the processed measurement data, based on the season and weather type specified by the user.

2. Get the number of non-zero samples ($n_{sp}$) in the production of $sp$ by subtracting the start index ($s_{sp}$) from the end index ($e_{sp}$). $s_{sp}$ and $e_{sp}$ is found using three different methods, and the latest start index and earliest end index of these are used. The three methods are as follows:

   (a) Set $s_{sp}$ and $e_{sp}$ to be the indices corresponding to the theoretically calculated times for sunrise and sunset in Konstanz, Germany at the day when the data was measured (where $AOI = 90^\circ$ for a fictional solar panel with tilt $\beta = 0^\circ$).

   (b) Set $s_{sp}$ and $e_{sp}$ to be the indices corresponding to the times when the sun enters and leaves the surface of the solar panels for which the data was measured (where $AOI = 90^\circ$ for the real PV panels used in Konstanz).

   (c) Set $s_{sp}$ and $e_{sp}$ to be the indices where $sp$ actually starts and ends its production, i.e. where the values go from zero to greater than zero and from greater than zero to zero.

3. Generate a theoretically optimal production profile ($opt$) with an initial sample time interval ($\Delta t_{init}$) for the given installation angles and rated output of the PV system to simulate, using the mathematical models for calculating AOI and IAM. $opt$ should only contain production after sunrise, before sunset, and when $AOI < 90^\circ$.

4. Find the number of minutes ($m_{opt}$) between the start and end of the production in $opt$. This is done by using the same method as in 2c for finding start and end indices, multiplying these by the interval $\Delta t_{init}$, and calculating the difference between them.
5. Calculate a new interval $\Delta t_{\text{new}} = \frac{m_{\text{opt}}}{m_{\text{sp}}}$ which is used to evenly distribute the samples in $sp$ over the time in $opt$.

6. Generate a new theoretically optimal production profile ($opt_{\text{new}}$) with the time interval $\Delta t_{\text{new}}$. This will be similar to $opt$, but with an interval that results in the same number of non-zero samples in the optimal production as in the measured production of $sp$.

7. Finally, for each production value between $s_{sp}$ and $e_{sp}$ in $sp$, multiply it by the corresponding values in $opt_{\text{new}}$ and store the values in a new profile. This is the production profile to be used for simulation.

The result of this process is a production profile that can be used in the simulation. It is based on a theoretically calculated optimal production for a given scenario, but with realistic variations applied to replicate weather fluctuations. It should be noted that this method retains the amount of variation in the production of the selected data profile, but modifies the time between each sample. This means that characteristics such as peaks and valleys in the production may be closer or further apart than they were in reality.

The reason why three different methods are used for finding the start and end of production in the selected profile in step 2 is that this is dependent on how the PV system was installed and its surroundings. Figure 9 shows a two-dimensional illustration of a PV panel and its surroundings, where $a_1$, $b_1$, and $c_1$ would be the potential start times and $a_2$, $b_2$, and $c_2$ would be the potential end times for the PV panel’s production, using the corresponding methods in step 2. In this specific scenario, $b_1$ and $c_2$ would be selected because they represent the latest start time and earliest end time.

Since method $c$ involves looking at where the production profile actually contains production values, and thus can detect any obstacles that delay or advance the start and end times, it can potentially be used in all situations. However, it is
likely that a few of the first and last values of the production profile comes from indirect solar radiation as a result of reflections, which is not considered in this context. Therefore, methods a and b are useful to make sure that the selected production period is only when the sun is able to radiate directly on the panel.

4.6 Indirect solar irradiance

Indirect solar irradiance is often a significant contributor to PV system’s output, as the solar radiation is reflected off of other objects such as clouds, lakes, and buildings. Twidell and Weir defined *direct beam* as the radiation observable from the direction of the sun, and *diffuse radiation* as the radiation observable from other directions [5, pp. 40-41]. It is also worth noting that these are observable from beneath Earth’s atmosphere, whereas the *extra-terrestrial beam* was defined as the solar radiation incident on the atmosphere from the direction of the sun, which despite small variations is considered to be constant for solar energy applications.

The total irradiance received by photovoltaic surfaces is the sum of direct beam and diffuse radiation. According to [5] the total irradiance received always consists of at least 10% diffuse radiation, even on days with clear skies, and up to 100% of the total irradiance is diffuse radiation on completely overcast days. Since the ratio between diffuse and direct radiation depends on many local factors, it is difficult to estimate as it varies significantly for different PV systems.

The measured energy production data used in this work contains the energy produced as a result of both diffuse and direct radiation, and the production displayed in the SolarSim application thus shows both. However, as described in Section 4.5, only the period when direct radiation is available has been considered, and diffuse radiation outside this period has been ignored. This was done to match the theoretically calculated optimal production, which only produces energy when $\text{AOI} < 90^\circ$ and the sun is over the horizon. As a result of this, and a limitation of SolarSim, no production is simulated when $\text{AOI} > 90^\circ$ or the sun is under the horizon, and this may sometimes create production profiles with a more abrupt start and end than one would typically see in real production profiles.
5 Methods for constructing energy consumption for simulation

5.1 Consumption data analysis

In order to construct the electricity consumption of individual household appliances that can be used for simulation, knowledge of how these appliances behave in the real world is necessary. However, this can be challenging as appliances are likely to behave differently depending on several factors. Taking washing machines as an example, these factors include the brand, the amount of clothes being washed, the mode or program that is running, and the water inlet temperature. Thus, running a washing machine at 40°C with a half-full load, may consume significantly less energy than running it at the same temperature with a full load.

During the CoSSMic project, energy meters were installed on several appliances in a selection of volunteer households in Konstanz, Germany. A large set of consumption data was stored, and the data was recorded as time-series with cumulative electricity consumption for each individual appliance over the full duration of the project. The data holds detailed information about behavior patterns for these specific appliances. However, comparing individual runs or cycles of an appliance is difficult when the data is in the form of long time-series. Therefore, as a part of this work, an algorithm has been developed for splitting the time-series into single, comparable load profiles for each appliance run. Furthermore, algorithms have been developed for classifying these profiles into clusters of runs, where each cluster represent an appliance program. Sections 5.1.1 and 5.1.2 describes how these two processes are implemented and Sections 5.2 and 5.3 shows their results on the washing machine and dishwasher data.

5.1.1 Time-series splitting algorithm

The splitting algorithm iterates over time-series data and analyzes whether an appliance is on or off. Generally, an appliance is considered to be off when the current data point has the same consumption value as the previous, i.e. where the cumulative time-series profile example in Figure 10 is flat. It is considered to be on when the current data point has a higher consumption value than the previous, i.e. where it increases. The algorithm then splits the time-series where an appliance
transitions between on and off, and vice versa, resulting in a collection of smaller consumption profiles. Each of these are presumably a single run or wash cycle of that appliance.

The algorithm first initializes two lists. The first list will contain the data values for the current run, and the second list will contain all runs. Then, the algorithm analyzes each data point in the time-series and appends the values to the current run when in on-periods. Each time the algorithm recognizes that the current run is done, that run is appended to the list of all runs, and the list with values for the current run is reset. Values in the off-periods contain no useful information in this context and are discarded. Finally, the individual runs are ‘reset’ so that they all start at 0 kWh, meaning that for each run the values are reduced by the first value of that run.

The algorithm is simple and effective. However, without any additional functionality, some noise or error may be included as appliance runs. This is presumably either due to data gaps, data measurement errors, or a lack of precision in the splitting algorithm. Removing this noise can be useful to better see the consumption patterns, as well as making classification of programs easier and more accurate. However, which runs should be considered noise may be difficult to decide, and it is important that correctly measured runs are not excluded as a result of trying to remove noise. Therefore, as an attempt to solve this, the following three threshold values are included as parameters in the algorithm. They do a good job of countering the specific issues found in the data set.

- **Maximum pause allowed**: The maximum duration of an off-period within a run. This allows small periods with no change in consumption, without concluding that the run is done.

- **Minimum duration**: The minimum duration of a run. All runs with a shorter duration than this value is considered to be noise, and is excluded.
- **Maximum consumption**: The maximum total consumption of a run. All runs with a higher total consumption than this value is considered to be noise, and is excluded.

Ideally, setting appropriate values for these parameters should prohibit any measurement or splitting errors to be included as appliance runs and negatively affect the classification of appliance programs. Sections 5.2 and 5.3 give details on what threshold values that have been used specifically for washing machines and dishwasher associated with the CoSSMic project.

### 5.1.2 Program classification using Mahalanobis distance

The data set from the CoSSMic project does not include any information about what type of modes or programs the appliances were run with. However, this information is relevant for simulation as it affects the consumption of the appliances. In order to assign appliance programs to the individual appliance consumption profiles, statistical classification of these profiles have been performed. This is to give the users of the simulation an option to choose specific programs for the appliances to be simulated. This choice is then reflected in the amount of electricity consumed and the appliances’ consumption behavior over time. Prior to classification, the set of profiles was initially divided randomly into two categories. Two thirds were labeled *training* data used for manual classification, and one third was labeled *verification* data used for automatic classification.

**Manual classification** of the training data involves arranging appliance run profiles into several groups (clusters). Each cluster should then contain profiles with similar characteristics and is defined to represent an appliance program. Specifically, this was done by first establishing a set of features that profiles can be compared by. For a profile of a washing machine or dishwasher run, two important features have been evaluated:

- Duration
- Total energy consumption

These two features work well for this purpose, since one appliance program presumably has about the same duration and consumes about the same amount of
energy every time, disregarding external factors. Thus, the manual classification of profiles was specified by setting minimum and maximum values for these two features for each cluster. Clusters were then assigned the profiles who’s features were within the specified minimum and maximum values.

It should be noted that these two features does not give the whole picture of a profile’s consumption behavior. Since its consumption development throughout the entire run may be of interest, using additional features that reflect this development may result in a more accurate classification. This is an interesting task worth considering in future work.

**Automatic classification** of the verification data involves assigning profiles to clusters by passing a certain criteria. In this work, the automatic classification has been based on evaluating the Mahalanobis distance between a profile from the verification data and the mean of the profiles assigned to a cluster. The Mahalanobis distance is a unit for measuring the number of standard deviations a point is from the mean of a distribution for multiple dimensions, first introduced in 1936 [18]. The Mahalanobis distance $MD$, as shown in [19], is given by Eq. (10)

$$MD = \sqrt{(x - \bar{x})C^{-1}(x - \bar{x})^T}$$

(10)

where $x$ is the feature vector for the point, $\bar{x}$ is the feature vector for the mean of the distribution, and $C^{-1}$ is the inverse covariance matrix. As shown in [20], the covariance matrix $C$ can be estimated using Eq. (11)

$$C = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(x_i - \bar{x})^T$$

(11)

where $n$ is the number of observations in the distribution and $x_i$ is the $i$-th observation.

As there are two features of the consumption profiles considered in this thesis, the vector for the mean of these two features for all profiles in a cluster was found, and the Mahalanobis distances from these means to the profiles of the verification data were evaluated. In general, if this distance was long, it was considered unlikely that the profile is of the same appliance program as represented by the cluster. Similarly, if the distance was short, it was considered likely that the run is of the
same appliance program.

Thus, a threshold value separating high and low Mahalanobis distances for each cluster is needed in order to determine whether or not a profile should be classified to that cluster. Here, the threshold was determined by the profile from the manually classified training data with the biggest Mahalanobis distance from the mean vector of the features of the profiles assigned to its cluster. This distance was considered the threshold for membership of a cluster, and profiles from the verification data was automatically assigned to clusters if they were within the threshold. Profiles with a distance higher than the thresholds of all clusters were left unclassified. Profiles within the threshold of multiple clusters were classified to the one with the shortest Mahalanobis distance.

The automatic classification of verification data shows the performance of the trained Mahalanobis threshold, and indicates its generalization, i.e. its applicability on ‘new’ data.

5.2 Washing machines

It is clear that washing machines are great tools that have changed the process of washing clothes from hard work to an easy, semi-automated household task. However, this task comes with the downside of consuming a significant amount of both water and energy. It is beneficial, both for individuals and for the society as a whole, to make this process more efficient.

Pakula and Stamminger conducted a comprehensive survey, based on previously reported data, comparing the energy and water consumption of washing machines in a wide range of geographical regions and with different technologies [21]. One of the conclusions that can be drawn from this survey is that there are large differences regarding the energy consumption of washing machines, not only over time, but also per wash cycle. Some of the reasons for this are reported to be different wash temperatures, wash duration, wash technologies, and wash habits and practices. For example, washing machines with vertical axis, common in North America, Australia, China, South Korea, and Japan, has a low energy consumption because they often use externally heated water from the tap. Washing machines with horizontal axis, common in Europe, use more energy because they have integrated heating of water, and they consume less water because only
a small portion of the wash tub is filled. Furthermore, people in regions such as Australia, Japan, and North America tend to have a higher number of wash cycles per year compared to Europe, primarily due to frequent use of short and cold washing programs.

As a result of the many factors and the complexity that determines the energy consumption of washing machines, as well as a lack of systematic measurements of consumption on specific programs, accurately estimating a washing machine’s consumption is not a trivial task. However, the average electricity consumption per wash cycle in Germany as reported by Pakula and Stamminger is 0.87 kWh, and similar values were reported for other European countries. This matches fairly good with the average consumption of wash cycles in the CoSSMic data (measured in Konstanz, Germany), and supports its validity. The data is thus more suited to be used for simulation of European washing machines than those from other parts of the world.

The predominant challenge regarding the data is the lack of labels for which washing machine programs that have been used. As an attempt to obtain this information, the data has therefore been split into individual cycles and classified into a set of programs.

5.2.1 Splitting time-series data

Using the algorithm for splitting time-series data described in Section 5.1.1, the consumption data for washing machines have been split into individual runs or wash cycles. To best remove noise and error from this specific data, the following parameters were considered suitable and used in the algorithm.

- Maximum pause allowed: 1 minute
- Minimum duration: 10 minutes
- Maximum consumption: 5 kWh

Figure 11 shows how the splitting algorithm has divided the time-series into individual wash cycle profiles, visually comparable by humans. There are significant variations in their total consumption, even within the same washing machine. Additionally, although some repeating patterns can be observed, fairly few clear
groups of cycles exist. The exact reason for this is unknown, but as previously
discussed, additional factors such as the amount of clothes may contribute to this
behavior.

As an example, for washing machine number 2 it is evident that most of the
cycles are very similar in duration and their overall shape, but varies significantly
regarding energy consumption. One could therefore assume that these cycles were
of the same washing machine program, and that the differences in energy con-
sumption were caused by differences in inlet water temperature or the amount of
clothes being washed. Additionally, a small group of shorter cycles with a run time
of approximately 50 minutes can be observed, suggesting that another program,
or a shorter version of the same program was being used. For the other washing
machines there is clearly a lower degree of consistency regarding the similarity in
shape across cycles.

5.2.2 Classification of washing programs

Classification of programs from the washing machine cycles has been performed
using the method presented in section 5.1.2. Figure 12 shows the distribution
of all individual wash cycles from the six washing machines associated with the
CoSSMic project as feature points representing their duration and total energy
consumption, the two features classification has been based on. By assigning a
unique color to each washing machine, the differences in their consistency of wash
cycle behavior becomes apparent.

Whether to focus on clusters separately for each washing machine or collec-
tively, was a choice that had to be made, as it would affect the classification
results. In order to perform a general classification that can yield good results
for most types of washing machines, the classification was done collectively here.
Figure 13 shows the washing machine cycles where a selection of two thirds of all
cycles were included as training data and the basis for manual classification. These
cycles were given the same color to easier see clusters across washing machines.
The manually set cluster parameters for minimum and maximum duration and
consumption values are illustrated with dotted lines. Next, the automatic classifi-
cation of the remaining verification data was performed. This is shown in Figure
14, where the successfully classified verification data is illustrated as green dots,
Figure 11: Cumulative energy consumption of individual wash cycles for washing machines 1-6 from the CoSSMic project, as a result of splitting time-series data.
5.2.3 Programs

Different types of washing machines often operate with different types of programs the user can select from. These typically have a specific water temperature, or are specifically designed for certain types of clothes. For simplicity, the programs defined to be used in the SolarSim application is only based on two of the most important factors that are common for all washing machines: water temperature and duration. SolarSim thus provides eight different program options: 30°, 40°, 60°, 90°, and a short version of each of these.

Since this number of programs does not correspond with the number of clusters obtained in the classification of washing machine cycles, a mapping has been done to link each program with a set of clusters. Thus, when a user of the simulator chooses a program, a random wash cycle profile from the corresponding clusters is selected. This mapping is shown in Table 3. It should be noted that this mapping was done manually with no exact knowledge of what type of programs the profiles...
Figure 13: Duration and total energy consumption of washing machine cycles from the CoSSMic project, where a selection of two thirds of the cycles are included as training data and the basis for manual classification. These cycles have been given the same color to easier see clusters across washing machines, and the manually set cluster parameters are illustrated with dotted lines.

Figure 14: Duration and total energy consumption of washing machine cycles from the CoSSMic project, where green dots show the successfully classified verification data and red dots the unsuccessful classifications.
Figure 15: Duration and total energy consumption of washing machine cycles from the CoSSMic project showing the final result of wash cycle classification, where each cluster is assigned a unique color. See Table 3 for the mapping of washing machine programs to these clusters.

actually derives from, and it is based on the assumption that wash cycles with high water temperature and long duration cause higher energy consumption than cycles with low water temperature and short duration.

5.3 Dishwashers

Automatic dishwashers are useful, and can save people a considerable amount of time. An additional advantage is that they also often save water and energy, compared to manual dish washing. In a survey by C. P. Richter, 200 households in four European countries were interviewed and observed, and the households with a dishwasher used 50% less water and 28% less energy on the dish washing task than the households without a dishwasher [22]. Richter also showed that further savings can be achieved by changing the user’s washing habits. Examples of this are that 20% of the assessed dishwasher baskets were only slightly filled and could hold more dishes, and that the amount of pre-rinsing of dishes could be reduced.

When it comes to the choice of washing program, Richter reported an average program temperature of around 59.3°C, and a great majority of users chose the same washing program every time. However, there are limited international data
### Program Clusters

<table>
<thead>
<tr>
<th>Program</th>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>30°</td>
<td>6, 9</td>
</tr>
<tr>
<td>30° short</td>
<td>1, 3</td>
</tr>
<tr>
<td>40°</td>
<td>10, 12, 14, 16, 17</td>
</tr>
<tr>
<td>40° short</td>
<td>2, 4</td>
</tr>
<tr>
<td>60°</td>
<td>13, 15</td>
</tr>
<tr>
<td>60° short</td>
<td>5, 7</td>
</tr>
<tr>
<td>90°</td>
<td>11</td>
</tr>
<tr>
<td>90° short</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 3: Mapping of washing machine programs to the clusters displayed in Figure 15

on this issue, and due to program differences between dishwasher brands and user’s presumably limited knowledge of which programs their dishwasher is equipped with, performing consumer polls can be difficult in terms of comparability [22].

Sections 5.3.1-5.3.3 describes how the dishwasher consumption data used in this work was split into individual washing cycles and classified into programs, similar to the process for washing machines.

#### 5.3.1 Splitting time-series data

Time-series data for dishwashers was split into individual cycles using the same method as for washing machines (presented in Section 5.1.1). To best remove noise and error from this specific set of dishwasher consumption data, the following parameters were considered suitable and used in the algorithm.

- Maximum pause allowed: 10 minutes
- Minimum duration: 10 minutes
- Maximum consumption: 3.5 kWh

Figure 16 shows the result of the splitting algorithm on the five dishwashers associated with the CoSSMic project. In general, the majority of the dishwasher cycles appear to have a total consumption of 0.7-1.5 kWh and a duration of 50-130 minutes. However, clear differences can be observed between the consumption patterns. For example, dishwasher 2 has two distinct consumption patterns, indicating that the user frequently switches between two different washing programs,
and that these programs have fairly consistent behaviour and total consumption every time. The same can be observed about dishwasher 3, except that the user seems to only use one washing program instead of two. Furthermore, although some patterns are present, the other dishwashers seem to have more inconsistency in the consumption. The reason for this is unknown, but possible explanations may include the user behavior, error in the time-series splitting, or error in the measurement data. It may also be caused by a frequent use of ‘auto’ programs, which varies the behavior of the wash cycle based on factors such as how dirty the dishes are.

### 5.3.2 Classification of washing programs

Classification of individual dishwasher cycles was done using the same method and features as for washing machines. Figure 17 shows the duration and total energy consumption of the dishwasher cycles, where each dishwasher is assigned a unique color. Although there are some variation, most of the cycles are distributed in distinct clusters, and the clusters are significantly more pronounced than for washing machines. As previously discussed, this is most likely due to the higher number of external factors that are relevant for washing machines.

Figure 18 shows the dishwasher cycles where a random selection of two thirds of all cycles were included as the training data and the basis for manual classification. The manually set cluster parameters for minimum and maximum duration and consumption values are illustrated with dotted lines. Furthermore, in Figure 19 the remaining verification data were automatically classified and assigned a cluster using the method of evaluating the Mahalanobis distance. Successfully classified cycles are shown as green dots and cycles that was not classified is shown as red dots. Finally, Figure 20 shows the result of the classification, and each cluster is assigned a unique color. Note that some outlier cycles were deliberately ignored in the manual classification, since these are likely to be a result of measurement error, or inaccuracy of the time-series splitting algorithm.

### 5.3.3 Programs

Although most of the clusters of dishwasher cycles are fairly clear and distinct, it is not obvious which clusters correspond to which washing programs. Additionally,
Figure 16: Cumulative energy consumption of individual wash cycles for dishwashers 1-5 from the CoSSMic project, as a result of splitting time-series data.
Figure 17: Duration and total energy consumption of dishwasher cycles from the CoSSMic project, where each dishwasher is assigned a unique color.

Figure 18: Duration and total energy consumption of dishwasher cycles from the CoSSMic project, where a selection of two thirds of the cycles were included as the training data and the basis for manual classification. These cycles are given the same color to easier see clusters across dishwashers, and the manually set cluster parameters are illustrated with dotted lines.
Figure 19: Duration and total energy consumption of dishwasher cycles from the CoSSMic project, where green dots show the successfully classified verification data and red dots the unsuccessful classification.

Figure 20: Duration and total energy consumption of dishwasher cycles from the CoSSMic project showing the final result of classification, where each cluster is assigned a unique color. See Table 4 for the mapping of dishwasher programs to these clusters.
Table 4: Mapping of dishwasher programs to the clusters displayed in Figure 20

<table>
<thead>
<tr>
<th>Program</th>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>economic</td>
<td>2</td>
</tr>
<tr>
<td>fast</td>
<td>1</td>
</tr>
<tr>
<td>delicate</td>
<td>3</td>
</tr>
<tr>
<td>normal</td>
<td>4, 5, 7</td>
</tr>
<tr>
<td>intensive</td>
<td>6, 8</td>
</tr>
</tbody>
</table>

different types of dishwashers may have different programs equipped, and may behave differently even though the same type of program is selected. For the purpose of providing intuitive options for dishwasher programs to the users of SolarSim, differences between individual dishwashers have been ignored and five common programs are used: economic, fast, delicate, normal, and intensive. Similar to the mapping of washing machine programs, a manual mapping of these programs to the dishwasher clusters has been done, and this mapping is shown in Table 4.

5.4 Background consumption

In this work, washing machines and dishwashers, and their consumption data have been analyzed and processed. These are examples of ‘shiftable’ appliances, meaning that their operating period can be shifted to another time to better balance demand and generation of electricity, without too much inconvenience to the user. However, it is safe to assume that a significant part of the total consumption in an apartment comes from non-shiftable appliances such as for heating, cooling, lights, etc. All consumption that does not come from washing machines or dishwashers is here referred to as background consumption.

Accurately estimating the background consumption of an apartment requires detailed information about all its appliances and their exact duration of ‘on-time’. Additionally, other factors such as the apartment’s insulation and the outside temperature is needed to determine the need for heating and cooling. While this is certainly possible, it can be difficult and impractical for users of a simulator to provide this information. Several internal and external factors also contribute to the complexity of this problem, such as the time of year, geographic location, electricity pricing, and the residents’ personal preferences.
One possible solution to this problem is to use statistical data on households' average energy consumption based on simple factors such as its size and number of residents, and use this to calculate a constant value for consumption of the specified apartments. However, this would likely give insufficient accuracy, primarily regarding the distribution of consumption throughout a day. As a result, estimation of background consumption has not been included in this work and is a relevant topic to investigate in future work.
6 Data Processing: software documentation

6.1 Overview

The data processing software is a set of modules developed as a part of this work. It implements the methods presented in the thesis for processing, categorization, and classification of household measurement data from the CoSSMic project. These methods have been used as preparation of the data used in the SolarSim application. Its main purposes are to split time-series data into smaller profiles for energy production and consumption, categorize production data, and classify consumption data. Specifically, energy production data from the PV system measurements are split into individual 24 hour profiles starting at midnight. It is then categorized into four seasons and four different weather types. For consumption data from washing machines and dishwashers, the time-series is split into individual wash cycles and classified into a set of washing programs.

Additionally, the software plots the processed data and creates visual figures. This provides helpful visualization of time-series, production and consumption profiles, the clustering process, etc.

The processed data has been included in SolarSim and when the users specify various configurations of time, weather, and appliances, the corresponding profiles or cycles are randomly selected and used to generate the simulation results.

6.2 Technologies and libraries

The data processing software is written with the Python programming language and consists of individual executable modules responsible for specific tasks. Data files are stored in Comma-Separated Values (CSV) and JavaScript Object Notation (JSON) files. Figures are plotted using the Matplotlib plotting library, which provides an interface for creating customized and advanced figures. As part of the classification of appliance consumption profiles, a function from Scipy for calculating the Mahalanobis distance has been used, specifically found under scipy.spatial.distance.mahalanobis.

---

16 JSON: https://json.org
17 Matplotlib: https://matplotlib.org
18 Scipy: https://scipy.org
6.3 Modules and structure

The data processing software consists of several modules responsible for specific tasks. In addition to these modules, some model classes are included to represent objects such as appliances or PV systems. It should be noted that several of the modules contain some duplicate functionality for easier execution. This is done so that each file is provided as stand-alone modules and can be run ‘as is’, without having to specify options for file locations, appliance names, etc. Tables 5 and 6 provide a short description of each of these modules and model classes. The modules are located in the root directory, and the model classes are placed under “./models/”. Note that the three modules for splitting time-series reads from the original data source presented in Chapter 3, and it is required that this data is placed in a folder “./data/original/”. When executed, the modules save processed data under “./data/edited/”, and figures under “./figures/”.

6.4 Source code

The source code for the data processing software is available online and can be found in the public GitHub repository vkvisli/DataProcessing\(^{19}\).

\(^{19}\)Data processing software repository: https://github.com/vkvisli/DataProcessing
<table>
<thead>
<tr>
<th>Modules</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TimeSeriesSplitting_WM.py</td>
<td>Performs splitting of time-series data for washing machines 1-6, saves individual wash cycle profiles in CSV files, and displays the profiles on visual charts.</td>
</tr>
<tr>
<td></td>
<td>TimeSeriesSplitting_DW.py</td>
<td>Performs splitting of time-series data for dishwashers 1-5, saves individual wash cycle profiles in CSV files, and displays the profiles on visual charts.</td>
</tr>
<tr>
<td></td>
<td>TimeSeriesSplitting_PV.py</td>
<td>Performs splitting of time-series data for photovoltaic (PV) systems 1, 3, 4, and 6 into 24 hour production profiles, normalizes them, and divides them into four seasons based on month number. Finally, it saves the profiles in CSV files, and displays them on visual charts.</td>
</tr>
<tr>
<td></td>
<td>Classification_WM.py</td>
<td>Performs manual classification on two thirds of the individual washing machine cycles using manually set clustering parameters, and automatic classification on the remaining third using the method of evaluating the Mahalanobis distance. The data is saved in JSON files. Finally, the clusters are displayed on visual graphs.</td>
</tr>
<tr>
<td></td>
<td>Classification_DW.py</td>
<td>Performs manual classification on two thirds of the individual dishwasher cycles using manually set clustering parameters, and automatic classification on the remaining third using the method of evaluating the Mahalanobis distance. The data is saved in JSON files. Finally, the clusters are displayed on visual graphs.</td>
</tr>
<tr>
<td></td>
<td>Categorization_PV.py</td>
<td>Categorizes the individual 24 hour PV production profiles into four weather types (sunny, partially cloudy, cloudy, and rainy). It saves the categorized profiles in JSON files and displays them on visual graphs.</td>
</tr>
<tr>
<td></td>
<td>ClusterParameters.py</td>
<td>This module provides the manually set cluster parameters for washing machine and dishwasher cycles and is not meant to be explicitly executed.</td>
</tr>
</tbody>
</table>

Table 5: Python modules for data processing of measured household data from CoSSMic
### Model classes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appliance.py</td>
<td>Represents a household appliance such as a washing machine or dishwasher, and provides relevant information and functionality.</td>
</tr>
<tr>
<td>PV.py</td>
<td>Represents a photovoltaic (PV) unit or system, and provides relevant information and functionality.</td>
</tr>
<tr>
<td>PV_Season.py</td>
<td>Represents a photovoltaic (PV) unit or system for one specific season, and provides relevant information and functionality.</td>
</tr>
<tr>
<td>Cluster.py</td>
<td>Represents a cluster of individual runs or cycles from an appliance.</td>
</tr>
</tbody>
</table>

Table 6: Python model classes for representing objects such as appliances or PV systems

## 7 SolarSim: software documentation

This chapter gives a detailed documentation of the developed SolarSim application, and provides descriptions of its purpose, how it works, how it has been deployed as a web application, and where to find the source code. The SolarSim web application\(^{20}\) is available online for all users.

### 7.1 Overview

SolarSim is a web-application that lets the users run simulations of the production and consumption of electricity in an apartment building with a shared PV system installed. The user specifies a set of parameters to represent either a real, planned, or fictional apartment building, such as the PV system’s rated output, its installation angles, and the configuration of individual apartments and their appliances. Appliances are given a certain time period to run rather than a specific time, so that they can be scheduled to start at any time within this period.

The goal of the simulation is to optimize the utilization of electricity from the solar panels, by scheduling the various appliances to run at the most beneficial times within the given time periods. This reduces the need for selling excess electricity to the grid, or storing it in batteries. Thus, the amount of electricity imported from the grid is also reduced.

\(^{20}\)SolarSim web application: [https://solarsim.appspot.com](https://solarsim.appspot.com)
The purpose of running simulations with SolarSim is to better understand how demand-side management can improve the utilization of solar energy, primarily in urban environments where several households share the same PV system. A major challenge with using solar energy in residential buildings is that most of the electricity is produced when people are away, e.g. at work or in school. By using smart appliance scheduling systems, some of the consumption can be shifted to periods where the demand is lower and the supply is higher, such as during work hours. However, it should be noted that this only applies for ‘shiftable’ appliances. These are appliances that can be started at varying times without any significant inconvenience to the user, such as washing machines and dishwashers.

SolarSim has been developed with a significant focus on visualization and user friendliness of the interface. It is built using components of Google’s Material Design, which is an “adaptable system of guidelines, components, and tools that support the best practices of user interface design”\textsuperscript{21}.

Figure 21 shows an example capture of the graphical user interface of SolarSim. It gives a good overview of the configured scenario and the configuration of apartments and their appliance runs. As opposed to writing the information in a

\textsuperscript{21}Material Design: \url{https://material.io}
command line or in input fields, the apartments are represented visually and users can easily configure them by clicking on various buttons and drop-down menus. This especially makes it easier to replicate real apartment buildings. Furthermore, icons are provided to indicate the meaning of various actions. It is easy to set up a desired scenario and run simulations on it, as very little knowledge of PV technology is required. The simulation results are displayed in a responsive chart with intuitive colors and explanations. Additionally, more advanced and detailed results are available for users who want to study the results more in-depth.

The targeted users of SolarSim is primarily professionals in the energy industry, researchers, and people involved with the planning of apartment building construction. However, it has been developed with the goal that anyone should be able to use it.

7.2 Running simulations

The scenario selector component provides the input fields for configuring the scenario parameters to be simulated, shown in Figure 22. Simulations are started by pressing the button labeled ‘start simulation’. This will run simulations using the selected scenario parameters, apartments, and appliance runs. Simulations can be started in either normal or repeat mode. Normal mode starts one simulation, while repeat mode indicates that the simulation is to be repeated a specified number of iterations using the same configuration. Repeat mode is selected by checking the box labeled ‘repeat’ and specifying a number of iterations. This is for example useful for obtaining confidence intervals of the average amount of energy needed to be imported from the power grid, due to higher consumption than PV production at any time of the day. Normal mode performs only one simulation and is useful because it
gives a more visual representation of the results in the form of a chart displaying information about the production and consumption profiles.

7.2.1 Scenario parameters

Scenario parameters are set by the user before starting a simulation and they are what defines the circumstances and environment of the simulation. A scenario consists of information about the time, location, weather, and the PV system. The following list gives a brief description of each scenario parameter.

- **Rated output at STC (kW)**: The PV system’s total rated electricity output at standard test conditions (STC) in kilowatt.

- **Vertical tilt**: The vertical tilt angle of solar panels in the PV system in degrees, where $0^\circ$ is horizontal and $90^\circ$ is vertical.

- **Horizontal orientation**: The horizontal orientation of solar panels in the PV system in degrees, where $0^\circ$ is south, $90^\circ$ is west, $-90^\circ$ is east, and $-180^\circ/180^\circ$ is north.

- **Weather**: The weather type to use in the simulation environment, affecting the amount of solar radiation received by the solar panels.

- **Start date**: The date used in the simulation. The current date can be used by checking the box labeled ‘Today’. Otherwise, a specific date can be selected.

- **Location**: The location used in the simulation. The user’s current location can be used by checking the box labeled ‘My location’. Otherwise, a specific latitude and longitude can be selected.

7.2.2 Apartments and appliance runs

Apartments and appliance runs are the electricity consumers in a simulation, and make up an entire apartment building. They can be configured by the user to match a specific building, as shown in Figure 23. It should be noted that an appliance run represent one run or cycle of an appliance, not the appliance itself. This is because one appliance can run several times within the simulation period.
The following list gives a brief description of the parameters that can be configured for apartments and appliance runs.

- **Apartments**:
  - **Number of apartments**: The number of apartments in the building.
  - **Appliance runs**: A list of appliance runs to be executed in an apartment.
  - **Number of people**: The number of people living in an apartment.
  - **Size**: The size of an apartment in m$^2$.

- **Appliance runs**:
  - **Appliance type**: The type of appliance for the appliance run (washing machine or dishwasher)
  - **Program**: The program of an appliance run.
  - **Start after**: The earliest time an appliance run can be started.
  - **Done by**: The time when an appliance run must be finished.

The electricity consumption of an apartment that does not come from washing machines or dishwashers is in this thesis considered background consumption, and is ignored. Therefore, the apartment parameters for the number of people and size, which can be used to estimate the background consumption, is ignored and does not affect the simulation. Nevertheless, these two parameters are included to illustrate how this information can be provided by the user, and utilized in future work.

### 7.2.3 Simulation results

When simulations are started using *normal* mode, the results of the simulation is displayed with visualization in focus. Production profile, consumption profile, and a profile for the sun’s angle of incidence is displayed in a chart. Additionally, a profile for estimated optimal production for the given scenario (regardless of weather type) is displayed for comparison. A cumulative version of these profiles can be enabled by checking the box labeled ‘cumulative’. The profile for angle of incidence, as discussed in Section 4.1, gives a clear indication of how the sun moves
throughout the simulation period relative to the surface of the solar panels. Depending on the scenario parameters, the angle of incidence will typically alternate between values close to 0° at noon and up to 180° at night. Furthermore, reference lines for the times of sunrise and sunset is provided. It should be noted that these are calculated theoretically based on the given time and location (see Section 4.3), and may therefore not always coincide with the actual times for sunrise and sunset.

For simulations started with `repeat` mode, the results are instead displayed as a list, with information about key aspects of the result from each simulation. For both modes, the option to download the results as a file is provided, by clicking the button labeled ‘save’. The results are then saved as a JSON file, and this file contains the most important result information, as well as the scenario, apartments, and appliance runs used. Figure 24 shows an example of the simulation results of a 10 kW PV system and 20 appliances.

### 7.3 Software architecture

The SolarSim web application consists of two main components, each developed as independent services: a graphical user interface (GUI) and an application programming interface (API). These are deployed in containers and hosted as services in Google Cloud. Details on the functionality, technologies, and deployment of
these services are provided in the next few sections. Figure 25 shows a high level overview of the software and deployment architecture for the SolarSim web application. In general, the GUI service provides the interactive graphical user interface to the client and the API service provides all technical functionality for creating the energy production and consumption to be used in the simulation, which is then visualized in the GUI. Both services are accessed by sending requests with the HyperText Transfer Protocol (HTTP).

As an integrated part of the API service, a previously developed appliance load scheduler software (see Section 7.5.2) has been included and is responsible for the scheduling of start times for the various appliances configured by the SolarSim user. The communication with the appliance load scheduler works by creating and exchanging CSV files internally in the API, containing the various production and consumption data.

### 7.4 Graphical user interface (GUI)

The SolarSim GUI service has been developed as a React application, and is primarily written with HyperText Markup Language (HTML), Cascading Style Sheets (CSS), JavaScript, and the JavaScript syntax extension JSX\(^{22}\). This service serves as the front-end for the SolarSim web application. React is a “JavaScript library for building user interfaces”\(^ {23}\), and is primarily based on populating an

\(^{22}\)JavaScript XML (JSX) syntax extension: [https://reactjs.org/docs/introducing-jsx.html](https://reactjs.org/docs/introducing-jsx.html)

\(^{23}\)React JavaScript library: [https://reactjs.org](https://reactjs.org)
application with encapsulated components, each managing their own state. This approach simplifies the complexity of large application, as it divides it into smaller and more manageable components with specific tasks.

When a user starts a simulation from the GUI, an HTTP POST request is sent to the SolarSim API, with information about the scenario parameters, apartments and appliances. The API handles all logic required to perform the simulation, and sends a response with the results when the simulation is done. The GUI processes the results and renders a visual representation of the simulation results.

In addition to the React framework, a few third party libraries and resources are used. These are listed in Table 7.

### 7.5 Simulation API

As part of the SolarSim software, the simulation API is a microservice dedicated to the functionality of simulating energy production and consumption, and serves as a loosely coupled back-end for the GUI. Its general purpose is to receive information via HTTP requests, containing necessary parameters of a simulation scenario, and return the simulation results as a response to the request.
<table>
<thead>
<tr>
<th>Name &amp; reference</th>
<th>Description</th>
</tr>
</thead>
</table>
| Recharts [http://recharts.org](http://recharts.org) | A library of react components for customizing and rendering data charts with SVG elements. This is used to display an ‘area chart’ for the simulation results.  
License: MIT |
| Material-UI [https://material-ui.com](https://material-ui.com) | A library of react components that implement Google’s Material Design. This is used to render elements such as buttons, input fields, icons, etc.  
License: MIT |
| react-responsive [https://github.com/contra/react-responsive](https://github.com/contra/react-responsive) | A media query module for responsive rendering of components. This is used to adjust how some components are displayed, depending on the device’s screen size.  
License: MIT |
License: Creative Commons (Attribution 3.0 Unported) |

Table 7: Third party libraries and resources used in SolarSim GUI
The API is developed as a Node.js application and is primarily written in JavaScript. The application starts a server using the HTTP module, which is responsible for handling requests and responses. The API is available online and endpoints for its resources are provided in Section 7.5.1.

The API is developed with the Representational State Transfer (REST) architectural style, which means that requesting systems can access it using a set of predefined stateless operations. It is primarily built as a part of SolarSim, but is openly accessible to anyone who wants to use it, and will accept requests with a valid format and payload from any source.

The random selection of production and consumption profiles from the pool of pre-processed profiles is implemented using JavaScript’s `Math.random` function. This function generates a pseudo-random number between 0 and 1, with an approximately uniform distribution.

### 7.5.1 Interface

The simulation API is accessed by sending HTTP requests to supported endpoints of its URL with a valid payload. For simplicity, it only accepts POST requests to the endpoint `/start`, which starts a simulation and returns its results. However, it is developed in a way that supports extension of endpoints and request types for additional functionality in future work. Table 8 gives an overview of the API interface, Table 9 provides documentation of parameters to include in request bodies, and Table 10 provides documentation of the parameters received as a response from the API. Finally, Table 11 shows an example of the request and response structure.

### 7.5.2 Appliance load scheduler

The appliance load scheduler (ALS) included in SolarSim is a simplified version of a scheduler that was originally developed for an energy smart neighborhood.

---

24 Node.js: [https://nodejs.org](https://nodejs.org)
25 Node.js HTTP module: [https://nodejs.org/api/http.html](https://nodejs.org/api/http.html)
26 SolarSim API: [https://api-dot-solarsim.appspot.com](https://api-dot-solarsim.appspot.com)
### SolarSim API interface

<table>
<thead>
<tr>
<th>Endpoint:</th>
<th>/start</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method:</td>
<td>POST</td>
</tr>
<tr>
<td>Request body:</td>
<td>A valid JSON object with the properties specified in Table 9 (see example in Table 11 for how to structure the object)</td>
</tr>
<tr>
<td>Response body:</td>
<td>A JSON object with the properties specified in Table 10 is returned as response to the request (see example in Table 11 for how the object is structured)</td>
</tr>
<tr>
<td>HTTP status code:</td>
<td>200 (everything went OK), 400 (invalid request body), 405 (method not allowed), or 500 (internal error)</td>
</tr>
</tbody>
</table>

Table 8: The SolarSim API interface

A microgrid with multiple PV producers in the CoSSMic project [14]. The simplified version used in SolarSim only considers one producer, i.e. the PV system configured by the user. Several repositories\(^{28}\) required to run the ALS have been included in SolarSim.

The start times for each appliance load is defined within its bounded time interval from the earliest start time to the latest start time. Note that the latest start time for and appliance load is found by subtracting the load’s duration from the ‘done by’ parameter. The ALS uses NLopt’s\(^{29}\) implementation of the Bounded Optimization BY Quadratic Approximation (BOBYQA) algorithm [23], that iteratively improves an initial random set of assigned start times for all appliances until the objective function no longer changes. The objective function to minimize is the net total energy imported from the electricity grid. Thus, as long as the energy consumption of all appliance loads can fit under the total PV production, the objective function is zero and it does not matter when they are started.

The integration between the Node.js application and the ALS within the API service consists primarily of creating and exchanging files in a shared location. When the API receives a request to start a simulation, the Node.js application creates CSV files containing energy production and consumption data, using the

---

\(^{28}\)DOMINOES: https://github.uiio.no/geirho/DOMINOES  
Theron++: https://github.com/GeirHo/TheronPlusPlus/tree/AMQ_Support  
CoSSMic: https://bitbucket.org/cossmic/taskscheduler/src/C++17%20Release  
LA-Framework: https://bitbucket.org/GeirHo/la-framework/src/default  
Optimization: https://github.uiio.no/geirho/Optimization  
CSV: https://github.com/ben-strasser/fast-cpp-csv-parser  
\(^{29}\)NLopt: https://nlopt.readthedocs.io/en/latest
<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scenario</td>
<td>(object)</td>
<td></td>
<td>An object containing the scenario properties</td>
</tr>
<tr>
<td>pvTilt</td>
<td>(number)</td>
<td>0 - 90</td>
<td>The vertical tilt angle of the solar panels in degrees (flat: 0°, vertical: 90°)</td>
</tr>
<tr>
<td>pvOrientation</td>
<td>(number)</td>
<td>-180 - 180</td>
<td>The horizontal orientation angle of the solar panels in degrees (South: 0°, West: 90°, East: -90°, North: 180/-180°)</td>
</tr>
<tr>
<td>pvCapacity</td>
<td>(number)</td>
<td>0 - 10000</td>
<td>The PV system’s total rated output at standard test conditions (STC) in kilowatt</td>
</tr>
<tr>
<td>unixTimeStart</td>
<td>(number)</td>
<td>0 - 10000000000</td>
<td>The start time (must be midnight) as a Unix time stamp in Coordinated Universal Time (UTC)</td>
</tr>
<tr>
<td>lat</td>
<td>(number)</td>
<td>-90 - 90</td>
<td>The latitude of the location to simulate</td>
</tr>
<tr>
<td>long</td>
<td>(number)</td>
<td>-180 - 180</td>
<td>The longitude of the location to simulate</td>
</tr>
<tr>
<td>timeZoneOffset</td>
<td>(number)</td>
<td>-12 - 12</td>
<td>The number of hours offset from Coordinated Universal Time</td>
</tr>
<tr>
<td>weather</td>
<td>(string)</td>
<td></td>
<td>The weather type to use for simulation (sunny, partially cloudy, cloudy, rainy)</td>
</tr>
<tr>
<td>apartments</td>
<td>(array)</td>
<td></td>
<td>An array containing apartment objects</td>
</tr>
<tr>
<td>id</td>
<td>(string)</td>
<td></td>
<td>A unique apartment ID</td>
</tr>
<tr>
<td>m2</td>
<td>(number)</td>
<td>1 - 1000</td>
<td>The apartment size in m²</td>
</tr>
<tr>
<td>nPeople</td>
<td>(number)</td>
<td>1 - 10</td>
<td>The number of people living in the apartment</td>
</tr>
<tr>
<td>applianceRuns</td>
<td>(array)</td>
<td></td>
<td>An array containing appliance runs</td>
</tr>
<tr>
<td>id</td>
<td>(string)</td>
<td></td>
<td>A unique appliance run ID</td>
</tr>
<tr>
<td>code</td>
<td>(string)</td>
<td></td>
<td>A code indicating type of appliance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>washing machine:</strong> wm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>dishwasher:</strong> dw</td>
</tr>
<tr>
<td>program</td>
<td>(string)</td>
<td></td>
<td>The program of the appliance run</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>wm:</strong> 30°, 40°, 60°, 90°, 30° short, 40° short, 60° short, 90° short</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>dw:</strong> economic, fast, delicate, normal, intensive</td>
</tr>
<tr>
<td>earliestStart</td>
<td>(string)</td>
<td>(00:00 - 23:59)</td>
<td>The earliest start time for the appliance run with the format “hh:mm”</td>
</tr>
<tr>
<td>doneBy</td>
<td>(string)</td>
<td>(00:00 - 23:59)</td>
<td>The time when the appliance run must be done with the format “hh:mm”</td>
</tr>
</tbody>
</table>

Table 9: Properties required in the JSON body when sending a request to the /start endpoint in the SolarSim API. See Table 11 for an example of how the request object should be structured.
<table>
<thead>
<tr>
<th>Property name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>production</td>
<td>(object)</td>
<td>An object containing the energy production</td>
</tr>
<tr>
<td>productionProfile</td>
<td>(array)</td>
<td>The simulated energy production profile in kW (power)</td>
</tr>
<tr>
<td>cProductionProfile</td>
<td>(array)</td>
<td>The simulated cumulative energy production profile in kWh (energy)</td>
</tr>
<tr>
<td>productionInterval</td>
<td>(number)</td>
<td>The number of minutes between each sample of the production profiles</td>
</tr>
<tr>
<td>productionProfileOpt</td>
<td>(array)</td>
<td>The theoretically calculated optimal production profile in kW (power)</td>
</tr>
<tr>
<td>cProductionProfileOpt</td>
<td>(array)</td>
<td>The theoretically calculated optimal cumulative production profile in kWh (energy)</td>
</tr>
<tr>
<td>productionIntervalOpt</td>
<td>(number)</td>
<td>The number of minutes between each sample of the optimal production profiles</td>
</tr>
<tr>
<td>aoiProfile</td>
<td>(array)</td>
<td>The theoretically calculated angles of incidence (AOI) in degrees</td>
</tr>
<tr>
<td>aoiInterval</td>
<td>(array)</td>
<td>The number of minutes between each sample of the profile for angles of incidence</td>
</tr>
<tr>
<td>sunRise</td>
<td>(number)</td>
<td>The estimated time of sunrise as a Unix time stamp in Coordinated Universal Time (UTC)</td>
</tr>
<tr>
<td>sunSet</td>
<td>(number)</td>
<td>The estimated time of sunset as a Unix time stamp in Coordinated Universal Time (UTC)</td>
</tr>
<tr>
<td>consumption</td>
<td>(object)</td>
<td>An object containing the energy consumption</td>
</tr>
<tr>
<td>consumptionProfile</td>
<td>(array)</td>
<td>The simulated energy consumption profile in kW (power)</td>
</tr>
<tr>
<td>cConsumptionProfile</td>
<td>(array)</td>
<td>The simulated cumulative energy consumption profile in kWh (energy)</td>
</tr>
<tr>
<td>consumptionInterval</td>
<td>(number)</td>
<td>The number of minutes between each sample of the consumption profiles</td>
</tr>
<tr>
<td>consumers</td>
<td>(array)</td>
<td>An array containing consumption information of each individual consumer</td>
</tr>
<tr>
<td>id</td>
<td>(string)</td>
<td>The unique ID for a consumer (corresponds with the IDs given for appliance runs in the request body)</td>
</tr>
<tr>
<td>assignedStartTime</td>
<td>(number)</td>
<td>The assigned start time for a consumer as a Unix time stamp in Coordinated Universal Time (UTC)</td>
</tr>
<tr>
<td>profile</td>
<td>(array)</td>
<td>The energy consumption profile of a consumer</td>
</tr>
</tbody>
</table>

Table 10: Properties provided in the JSON response body when sending a request to the `/start` endpoint in the SolarSim API. See Table 11 for an example of how the response object is structured.
Table 11: Example of the structure of JSON request and response objects for the /start endpoint in the SolarSim API
methods presented in Chapters 4 and 5. Next, a command is issued, telling a new instance of the ALS to be executed. The ALS calculates the optimal start time for each consumer based on the specified CSV files, and a new file containing the start times is created. This file is then used by the Node.js application to create the final production and consumption profiles which is returned as a response to the HTTP request sent by the client. Finally, the created files are deleted to reduce the disk storage.

To allow several clients to use the API simultaneously, a unique ID for each request is appended to the file names of the created CSV files, and a new instance of the ALS is executed for each request.

7.6 Deployment and hosting

The SolarSim application has been deployed and is hosted on Google App Engine (GAE)\(^{30}\). This is a platform that provides deployment and hosting of software applications on Google’s cloud servers, with automatic scaling and ‘pay for what you use’ billing. Deployment is based on using containers, which run the applications in preconfigured, secure, and reliable environments. GAE offer two main approaches of deploying code, using the standard or flexible environment. If the standard environment is selected, the user only specifies one of GAE’s supported programming languages or runtimes, and the container will be configured automatically. If a non-supported language is used, or a specific configuration is needed, the flexible environment allows the user to write a detailed description of the container using a ‘Dockerfile’. In both cases, the application’s environment, language, and runtime must be specified in a file called ‘app.yaml’.

Google App Engine structures an application and its underlying components in a hierarchy, as shown in Figure 26. An application can have several services, each with several different versions of that service. Furthermore, several instances of each version can be served. If automatic scaling is enabled, the number of instances will be adjusted to match the current user traffic. In the SolarSim application there are two services, GUI and API, both with one running version each.

\(^{30}\)Google App Engine: [https://cloud.google.com/appengine](https://cloud.google.com/appengine)

7.6.1 Deployment of the GUI

As described in Section 7.4, the SolarSim graphical user interface (GUI) is written in JavaScript using the React framework, and it runs in the Node.js runtime. Since Node.js is one of Google App Engine’s supported runtimes, the GUI is deployed using the standard environment and the container to run the application is configured automatically. The source code includes the file ‘app.yaml’, which specifies the environment and runtime used.

7.6.2 Deployment of the API

Similar to the GUI, the API has been deployed as a service on Google App Engine. However, the appliance load scheduler (see Section 7.5.2) is written in C++, which is not one of Google App Engine’s supported languages. Therefore, a custom Docker\(^{32}\) container has been configured to run both Node.js and C++ code and is deployed in the flexible environment. Docker is a tool for packaging standardized units of software into reliable, lightweight containers that include everything the software needs to run.

The custom container used to deploy the SolarSim API uses the Linux operating system and is based on Docker’s official image for the Fedora distribution\(^{33}\) (version 29). On top of this, a set of libraries required to run the appliance load scheduler is installed, and a new image with these changes was created. This image then serves as the base image for building the SolarSim API container, and

---

\(^{32}\)Docker: [https://www.docker.com](https://www.docker.com)

\(^{33}\)Docker official image for Fedora: [https://hub.docker.com/_/fedora](https://hub.docker.com/_/fedora)
is available online on Docker hub as solarsim_api, along with a description of which additional libraries have been installed.

When deploying the API application code to Google App Engine, the container was built using the configuration specified in the included Dockerfile. This process includes starting with the base image (solarsim_api), copying all application code into the container, compiling the appliance load scheduler, starting the JavaScript HTTP server, and finally exposing port 8080. Once built, the container was started and the server listens for incoming requests on port 8080.

7.7 Source code

The source code for the SolarSim GUI and API services is available online and can be found in the public GitHub repositories vkvisli/SolarSimGUI and vkvisli/SolarSimAPI.

The main functionality of the GUI service is located in the src folder, in which the React components, JavaScript models, images, and CSS styling are found in the folders components, models, images, and styles respectively.

For the API service, the main functionality is located in the root directory. JSON files containing the pre-processed energy production and consumption profiles are placed in the folder “./data/”. The files required to run the appliance load scheduler is located in the folder “./simulator/”, and this is also the location where CSV files for communication between the main API functionality and the scheduler are temporarily created.

---

34Docker image used to deploy SolarSim API: https://hub.docker.com/r/vkvisli/solarsim_api
35SolarSim GUI repository: https://github.com/vkvisli/SolarSimGUI
36SolarSim API repository: https://github.com/vkvisli/SolarSimAPI
8 Discussions and future work

8.1 Additional appliances

There are several topics that are interesting to investigate as potential tasks to look further into in future work. An example is that this thesis have focused on washing machines and dishwashers as the consumers for simulation of energy consumption. Other types of household appliances can be evaluated in future work. The charging of electric vehicles may be especially interesting to investigate in this context, as this task often have a high potential for saving energy by shifting the consumption to better suited times of the day. Additionally, models for estimating an apartment’s background consumption for the appliances that can not be scheduled can be studied, in order to simulate its total consumption, rather than the consumption of a selection of appliances. Such models will most likely be based on the number of residents and apartment size, and presumably additional, more complex parameters.

8.2 Machine learning for program classification

The use of various machine learning techniques as an alternative approach for classification of appliance consumption profiles into programs can be evaluated and compared with the statistical approach of evaluating Mahalanobis distances, which is used in this work. If machine learning techniques can outperform this approach by considering the entire time-series instead of only their duration and total energy consumption, more correct classification of appliance programs can potentially be made. This is for example because two different programs may in some cases have the same duration and consume the same amount of energy, but have very different consumption behavior during the run. As an example, Hüsken and Stagge has reported to achieve good classification of time-series using recurrent neural networks (RNN), as it makes efficient use of temporal information in the input sequence [24].
8.3 Battery storage

Saving excess energy from solar panels is an essential part of many PV systems and can be explored in future work. In the context of this thesis the use of shared, building level batteries can be studied in order to establish methods of simulating this type of storage in software. Simulation of energy storage would have to rely on models for estimating their efficiency, which has been reported to be between 70% and 95% depending on the manufacturer and battery chemistry [2].

8.4 Indirect solar irradiance

Models for estimating the amount of indirect solar irradiance on PV surfaces can be evaluated in order to simulate energy production in periods when the sun is not directly radiating on the surface. This is presumably a difficult task as it depends on local factors such as nearby mountains, lakes, buildings, and the cloud cover.

8.5 Representative data sets

The energy production and consumption data used in this work was measured in Konstanz, Germany. As a result, it is likely to contain production patterns that are typical for the weather and climate of this specific location. Additionally, it presumably contains consumption patterns that are typical for the human behavior and the types of appliances that are common in this location. Studying data sets from other countries or continents is therefore a relevant task for future work, and this can potentially better represent other climates and human behaviors, and contribute to creating more realistic simulations for these locations.
9 Conclusions

The thesis has presented challenges and requirements for simulation of shared PV energy production and appliance consumption scheduling in apartment buildings. Factors that influence the production and consumption of energy has been introduced and methods for constructing production and consumption profiles that can be used in simulation have been presented. The energy production profiles are constructed using a combination of mathematical models for calculating how the sun’s angle of incidence on PV panels affect the production, and the weather variations in real measurement data from selected households in Konstanz, Germany. The energy consumption profiles for washing machines and dishwashers are constructed by extracting and replicating consumption data from these households and classifying it into several types of washing programs.

As a result of implementing the presented methods, the simulation software called SolarSim has been developed and a technical documentation has been provided. SolarSim is a web application that lets users configure various scenarios and simulate the production of electricity from a shared PV energy system, and the energy consumption of washing machines and dishwashers in apartment buildings. It has been equipped with an existing appliance load scheduler and aims to optimally schedule the appliances to start at various times of the day within a given period, based on the availability of solar energy throughout that day.

The research hypothesis for the thesis was that real energy data measured at a specific location can be used to simulate production and consumption of energy for any apartment building at any time and location, by performing suitable pre-processing and combining it with mathematical models applied to parameters specific for the environment to be simulated. The conclusion from this work is that this is possible using the methods and models presented in the thesis, and that such simulations can contribute to making decisions about whether to invest in shared PV systems in apartment buildings. However, location-specific energy measurement data is often influenced by the prominent factors for that location, such as the climate and culture, and does not perfectly represent the factors for other locations.
References


