Manfred Rabl-Pöchacker

Modeling and Simulation of Smart Microgrids and Power Profiles

DOCTORAL THESIS

submitted in fulfilment of the requirements for the degree of

Doktor der technischen Wissenschaften

Alpen-Adria-Universität Klagenfurt

Fakultät für Technische Wissenschaften

Mentor
Univ.Prof. Dipl.-Ing.Dr. Wilfried Elmenreich
Alpen-Adria Universität Klagenfurt
Institut für Vernetzte und Eingebettete Systeme

Evaluator
Univ.Prof. Dipl.-Ing.Dr. Wilfried Elmenreich
Alpen-Adria Universität Klagenfurt
Institut für Vernetzte und Eingebettete Systeme

Evaluator
Prof. Dr. Ettore Bompard
Politecnico di Torino
Dipartimento Energia

Klagenfurt, December 2016

Affidavit

I hereby declare in lieu of an oath that

- the submitted academic paper is entirely my own work and that no auxiliary materials have been used other than those indicated,
- I have fully disclosed all assistance received from third parties during the process of writing the paper, including any significant advice from supervisors,
- any contents taken from the works of third parties or my own works that have been included either literally or in spirit have been appropriately marked and the respective source of the information has been clearly identified with precise bibliographical references (e.g. in footnotes),
- to date, I have not submitted this paper to an examining authority either in Austria or abroad and that
- the digital version of the paper submitted for the purpose of plagiarism assessment is fully consistent with the printed version.

I am aware that a declaration contrary to the facts will have legal consequences.

Signature	Date/Place

Acknowledgement

This page offers space to thank those who helped in any way to finalize this thesis in hand. When looking back the way I wonder about the curves and turn-arounds I went through. After reaching the final point I reason that it was more about the way to go and the growth on the path than about the point to reach. Many persons contributed to enrich my life while walking the PhD-candidate way. I am graceful to all of them, and some are mentioned individually in the following.

Obviously my adviser Wilfried Elmenreich took a central role during the last years. He made my PhD at Lakeside Labs possible by hiring me, supporting me in many different ways and I highly appreciate the possibility to participate at his experience and share his enthusiasm on computer science and engineering. I thank him for being available at any time and for putting the necessary resources to make my PhD thesis a success.

I like to thank all my colleagues at the Institute of Networked and Embedded System of the Universität Klagenfurt, especially Dominik Egarter, Andrea Monacchi and Sergii Zhevzhyk, for creating a friendly working environment, for sharing experience, coffee breaks, and some leisure hours with me. Special thanks go to Lizzie Dawes and everyone else who helped in English proofreading. Lizzie contributed essentially to improve the experience of all the future readers of this dissertation.

I conclude by expressing my gratefulness to my parents, Veronika and Johann Pöchacker, and my family who supported me, trusted in me and my capabilities, even when I was less sure about. Finally, I like to mention the most important person during my thesis writing period. She supported me in every possible way and she was patient and motivated me, whichever was required at that point in time. For this reason I dedicate this doctoral thesis to my beloved wife Veronika Rabl. I am happy to share such a big part of my life with her.

Abstract

The provision of energy, in particular the availability of electrical energy, is critical for societies worldwide to function. The demand for energy globally is ever increasing and thus every effort should be made to make the supply of energy greener and more sustainable. Smart grid technology has an essential role to play in this respect, for example, by enabling more efficient grid control and by better integrating renewable energy sources. The high share of the injected power by renewable energy require alternative control mechanisms for the power grid. Smart microgrids are a promising concept for the structuring of the power grid of the future. Grid simulation allows to explore the opportunities and drawbacks of microgrids as well as examining the hypothetical grid topology and infrastructure. Simulation allows potentially dangerous scenarios to be tested without putting costly hardware at risk. Although simulation of transmission networks, particularly AC power flow analysis, has been widely investigated, the simulation of distribution networks and microgrids remains at an early stage.

The first part of this dissertation contains a general model for smart grids, with respect to the flow networks of electrical energy, information, and payment. The flow network model serves as a reference model for the subsequent evaluation of current open-source software for power system simulation. The results for four simulators, used to simulate the IEEE 14-bus test case, will be presented. It will become apparent that established software for power grid simulation lacks the capacity to simulate renewable energy sources, as well as the residential demand for individual households. The second part of this body of work is concerned with RAPSim (Renewable Alternative Power System Simulation), one recently developed software tool for smart microgrids, which supports models for renewable energy generation. RAPSim has an easy-to-use graphical user interface and offers opportunities for extensions, but also for user-specific modifications, which will also be described in this work. RAPSim's capabilities are demonstrated using two test cases, including the IEEE 14-bus test case. Finally, areas for further development are outlined and the case is made for RAPSim's suitability for smart grid simulation.

Load disaggregation or non-intrusive load monitoring are efficient ways to identify which devices are operating in a given set of devices. Techniques of this kind can allow small-scale users, such as individual households, to play an active role in the smart grid. Data for both real-world and simulated consumption profiles are used to assess the performance of load disaggregation, where certain power consumption cases present particular difficulties. Recently, there have been attempts to quantify the difficulties presented by different consumption profiles, without the use of load disaggregation. In the third part of the dissertation, such a measure for consumption profiles is proposed. The measure, named as the proficiency of power values, is based upon entropy values and is motivated by information theory. The relevance of proficiency will be highlighted by comparing eighteen real-world device sets, which have been measured on different measurement campaigns of household power consumption.

Kurzfassung

Die zuverlässige Energieversorgung, insbesondere die Verfügbarkeit von Elektrizität ist weltweit zur Notwendigkeit in jeder Gesellschaft geworden. Der globale Energieverbrauch steigt stetig an und somit bedarf es dringender Bemühungen Energieversorgung nachhaltiger und umweltschonender zu gestalten. Smart-Grid Technologien können dazu einen wesentlichen Beitrag leisten, beispielsweise durch effizientere Netzsteuerung oder verbesserte Integration erneuerbarer Energieträger. Der hohe Anteil der Einspeiseleistung durch erneuerbaren Energiequellen verlangt alternative Regelmechanismen für das Stromnetz. Smart Microgrids sind eines der vielversprechenden Konzepte zur Strukturierung zukünftiger Stromnetze. Mittels Simulation können die Möglichkeiten sowie erwartbare Schwierigkeiten von Microgrids ausgelotet werden. Sowohl hypotetischer Netzausbau als auch potentiel gefährliche Szenarien können getestet werden ohne kostspielige Anlagen zu gefährden. Während Simulationen von Übertragungsnetzen, im Speziellen Lastflussanalysen, weit verbreitet sind entwickelt sich die Simulation des Verteilnetzes, als auch von Microgrids, erst seit kurzem.

Der erste Teil der vorliegenden Dissertation beinhaltet ein allgemeines Model für Smart Grids, welches durch Betrachten der resultierenden Netzwerke des Flusses von elektrischer Energie, Information und Zahlungen entsteht. Das Flussnetzwerkmodel dient als Referenz für die folgende Zusammenfassung aktueller Open-Source Software zur Simulation von Energieversorgungsnetzen. Für vier der Simulatoren werden die Ergebnisse der Lastflussanalyse für den IEEE-14-bus Testfall präsentiert. Weiters zeigt sich, dass es gängiger Software für Stromnetze an Simulationsmöglichkeiten für erneuerbare Energiequellen und von Verbrauchsprofilen einzelner Haushalte mangelt. Der zweite Teil der Arbeit widmet sich RAPSim (Renewable Alternative Power System Simulation), einem kürzlich entwickelten Softwaretool zur Simulation von Smart Microgrids, welches auch die Modellierung erneuerbarer Energiequellen unterstüzt. RAPSim bietet eine einfache grafische Benutzeroberfläche und Möglichkeiten zur Erweiterung und Benutzeranpassung, welche hier auch beschrieben werden. Die Möglickeiten von RAPSim werden anhand zweier Testfälle, einschließlich dem IEEE-14-bus Testfall, demonstriert. Abschließend werden mögliche Entwicklungsfelder aufgezeigt und die Eignung von RAPSim zur Simulation von Smart Microgrids wird überprüft.

Load Disaggregation, oder Non-intrusive Load Monitoring sind effiziente Verfahren um die laufenden Geräte in einem bekannten Set von Geräten zu bestimmen. Derartige Techniken können kleinen Verbrauchern, so wie einzelne Haushalte, die aktive Teilnahme am Smart Grid ermöglichen. Um die Funktion von Load Disaggregation zu demonstrieren werden sowohl gemessene als auch simulierte Lastprofile verwendet, wobei manche Lastfälle spezielle Schwierigkeiten mit sich bringen. Seit kurzem gibt es Bemühungen diese Schwierigkeit von Lastprofilen zu quantifizieren ohne tatsächlich Load Disaggregation ausführen zu müssen. Im dritten Teil dieser Dissertation wird ein derartiges Maß für Lastprofile vorgeschlagen. Diese, Proficiency von Lastwerten, ist eine Art Entropie und wird informationstheoretisch motiviert. Die Bedeutung von Proficiency wird anhand von Gerätesets aus drei verschiedenen Messkampanien für Verbrauchsdaten demonstriert.

Contents

List of Acronyms

AI Artificial Intelligence

CHP Combined Heat and Power

DER Distributed Energy Resources

DR Demand Response

EV Electric Vehicle

GUI Graphical User Interface

ICT Information and Communication Technology

IT Information Technology

NILM Non Intrusive Load Monitoring

MG Microgrid

PV Photovoltaic

RAPSim Renewable and Alternative Power System Simulation

SCADA Supervisory Control and Data Acquisition

SGAM Smart Grid Architectural Model

List of Figures

List of Tables

CHAPTER 1 Introduction

"Status quos are made to be broken."

– Ray A. Davis

The modernization of the electrical power system facilities is progressing on all levels. While large scale power plants and transmission grids are widely equipped with Supervisory Control and Data Acquisition (SCADA) systems it is only recently that the smart communication technology has penetrated power facilities also on smaller scale, such as in distribution systems and beyond. Distribution systems were poorly monitored as they were mainly passive which will no longer be the case in future. In addition to advances in Information and Communication Technology (ICT) the emergence of small and distributed power generation units will enhance development toward a more distributed and smarter meaning logically connected power grid. An expected benefit from the integration of ICT with the grid is that the investments into grid infrastructure can be saved due to more efficient control. For several reasons, Microgrids (MGs) [Cho09, p9] are a promising concept for grid re-organization and modernization. Supply security can be increased by MGs as they are able to run in island mode. Further they can increase local autonomy, which is a key motivation for private investment in small scale production facilities, like rooftop PV systems. At the same time proper integration of MGs is required for smooth interaction with the wider transmission system.

1.1 Smart Microgrid Simulation

Traditionally, MGs have been applied for literal island and onboard vehicles, as for instance aboard ships. Their development towards a possible universal unit or cell for power grids needs additional effort on economic and legal as well as technical levels. As not all of the related research questions can be answered by trial and error in exemplary utilities, software-based simulation is the method of choice for many research questions. Such "virtual" experiments do not obviate demonstration projects

and prototype creation but can realize tasks and answer questions faster and more economically. It would be useful if grid operators and electricity producers, as well as all possible electricity prosumers were able to access simulation software and relevant data. It is foreseeable that easy performable simulation and availability of meaningful and suitable data would enhance smart (micro) grid development on all levels. Further, reliable data are also essential for simulation models and many related engineering problems. Many homeowners have PV systems installed and are potentially interested in assessing their performance. The simulation of possible scenarios is also necessary to design appropriate control strategies for MGs. Those strategies must eventually be tailored to the specific conditions (time and place). In general, a scenario can be seen as a power demand and power generation time function that must be matched. The necessary integration of renewable energy sources and provision of future just-in-time and just-in-place [Ili08] energy services requires more information in the fastest possible time.

Consequently the MGs feature list [Cho09, p58] is a diverse one, and so too is the list of necessary research. It contains off- and on-grid production scheduling, possibly load-drop, storage management, prognosis of renewable production, demand forecast, grid synchronization, several safety and security features, and so on. There are also important economic questions to answer that might determine which business model (if any) will be successful with MGs in the long term. Essential changes in the power business are not likely to be driven by those who are established and successful within the traditional grid structure. These individuals understandably would not want to share or lose revenue. Participation in the current energy wholesale market, triggered in fifteen minutes intervals, is unrealistic for MGs as most of them do not reach the capacity limit. This has led to the emergence of aggregators that merge several small scale production facilities of MGs, allowing them to participate in the wholesale power market. This is where concepts such as virtual power plants begin to appear. Some of them aim for the control energy market with higher margins also. However the inclusion of Distributed Energy Resources (DER) is and will be organized economically, i. e., on the business level, they need controllers at the single source and also at the aggregated level. Those will need to deal with renewable production and (residential) demand which can be affected by somewhat unpredictable fluctuations in short timescales. Controller development is an example of where appropriate software simulation tools are required.

Power system simulation has already been performed by early computers to estimate load distribution within a meshed AC-grid (known as the power flow problem). Being a non-linear multidimensional problem, it can still be challenging, especially for large systems. Since that early days power system simulation developed into different fields dealing with electromagnetic phenomena on different timescales. Figure 1.1 from [Mil10, p14] relates different timescales to the phenomena and control strategies. There exist many commercial software tools as well as some open source programs for the various fields. The requirements for the simulation of MGs match those of larger power systems in many cases. Establishing a MG brings with it a number of specific requirements.

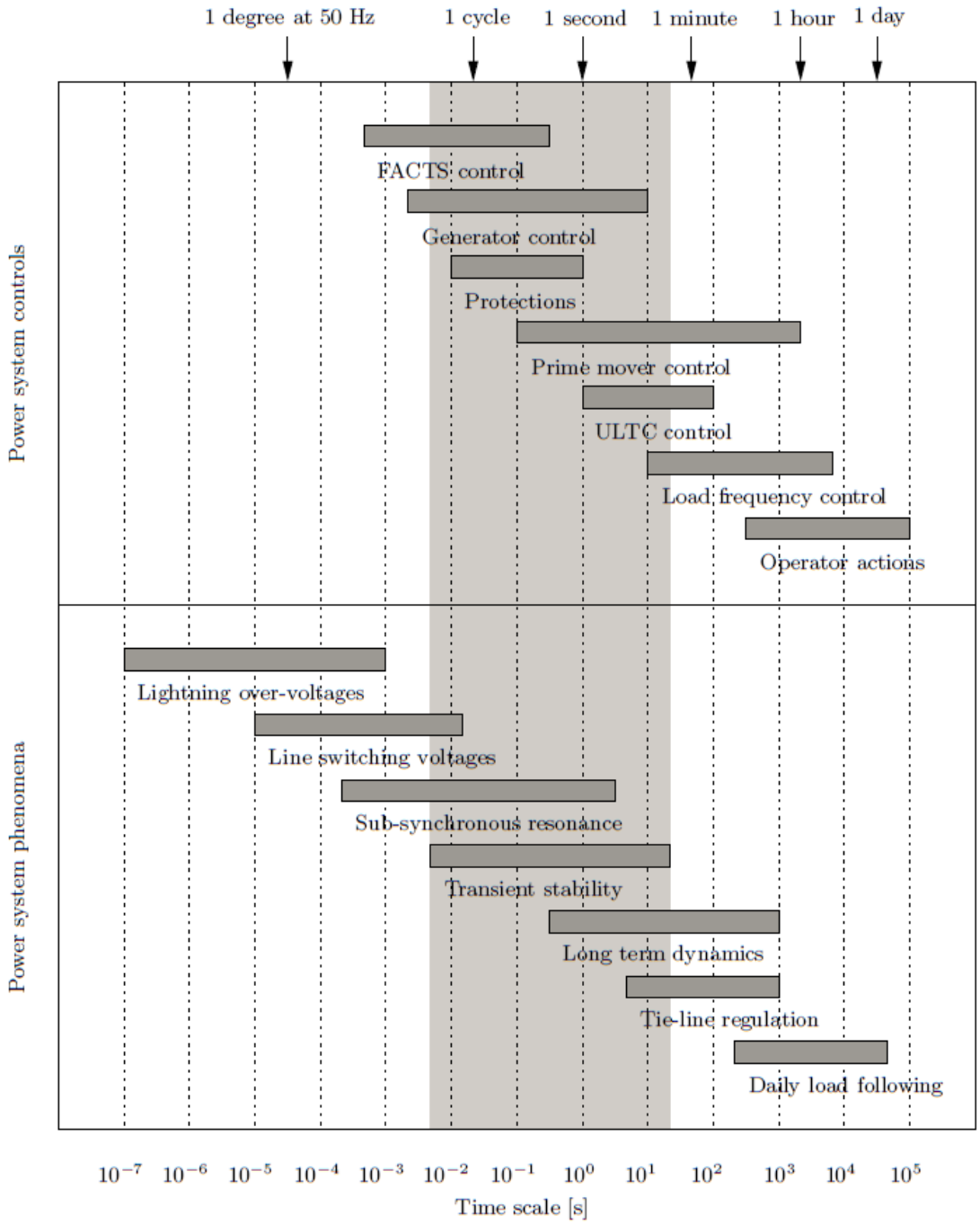

Figure 1.1: Time scales of power system phenomena and related controls according to [Mil10]. The gray area is covered in the cited book.

These requirements vary by the included types of renewable production, consumer characteristics, availability of reference cases, etc. Due to the smaller size of MGs statistical models are rarely as applicable as for transmission grid simulation. For

instance, if too few households are involved, the downscaling of the average residential load curves is no longer meaningful. Effects of statistical averaging do not occur if PV panels are situated closely to one another and exposed to the same weather conditions. Therefore, weather and climate data for a specific location allows all the simulated renewable generators to be fed with the same data, especially when all facilities are expected to be located close to each other. The weather itself is highly dependent on the geographical location and the specific period of the year.

For simulation of single components in the MG the availability of data is highly desirable. Beside other applications data can be used to inform engineering decisions, like sizing of a storage system, or to design models for other simulations. Many modern controllers working with Artificial Intelligence (AI) depend heavily on suitable data. Some energy consumption dataset that were collected in different research projects are publicly available, like [Mon14, Kol11, Bec14]. Their use in research can be limited by their range of applicability or when the conditions in which the measurements were taken are not apparent. In addition to the data itself, it would be useful if there were a method of comparing the quality of datasets. Load disaggregation or NILM is such an application where measurement based data are required within development and testing. In the special case of NILM this thesis investigates the problem of data comparability and proposes measuring the proficiency of power values as an approach for addressing the problem.

1.2 Load Disaggregation

There are several reasons why it is beneficial for a power grid operator to obtain as much information as possible in order to accomplish monitoring and controlling purposes, like giving consumption feedback, or detecting devices with high energy consumption. To avoid additional hardware, installation, and operational costs, it is highly valuable to derive this information from few, if not a single, measurement points. This applies in particular to residential consumers where the return on investment is low. Load disaggregation or Non Intrusive Load Monitoring is a technique used for reasoning about the operation of power consuming devices from a single measurement point recording the total power draw. One promising application of NILM is metering within smart homes [Per10], where information about single appliance usage is of high interest but monitoring with many sensors is not an option. Low-cost power monitoring on the device level is one step in integrating residential buildings into the future smart grid, which is considered to be a key technology for reducing carbon emissions.

NILM was first introduced by Hart in [Har92]. Even though current methods might use different device parameters, the original approach of decoding power values is still current. A variety of related studies rely on real worl measurement data of individual households´ power consumption on the device level. Such datasets are a valuable resource to calibrate or evaluate mathematical models by offering data from real world use cases.

However, it would be helpful to have methods to compare different use cases according their difficulty or complexity. Calculation of proficiency can be executed by using exclusively the device sets properties, whereas for a success rate-based approach it is necessary to execute an algorithm.

1.3 Problem Statements

The development of smart grids and MGs requires proper software simulation tools. Simulation is an indispensable method for research and development on the way to combine ICT and power grid engineering. Alongside the technological challenges are economical, ecological and social changes and requirements that should be considered by the choice of the simulation platform. An essential feature is easy accessibility. This includes open software standards and intuitive usability for non-programmers. The first two problem statements refer to that field and are:

A) What features do the currently available grid simulation software tools provide? Is there an open source smart grid simulation tool that covers the additional aspects for smart microgrids?

B) Renewable and Alternative Power System Simulation (RAPSim) is a newly developed software tool for MG simulation. What features does it currently provide and what further functions are required?

In addition to other technologies, Load Disaggregation or Non Intrusive Load Monitoring is a technology that requires real life data to evaluate performance as well as to train AI. Performance evaluation requires different problems that are specific enough for any of the solution methods and general enough to make sense with all the different approaches.

C) How can an information theory approach be helpful to assess the problem difficulty of load disaggregation by power values?

1.4 Structure and Content

This chapter gives a general introduction into the topics of smart microgrid simulation and load disaggregation before the problems tackled in this thesis are outlined. Chapter 2 has more detailed background information and contains basic details of related topics. A reader who is familiar with the field is not required to read this chapter to understand the three chapters which follow it. Chapters 3, 4, and 5 contain the main contributions and can be read individually. They have their own introduction sections and also give individual conclusions.

Within chapter 3 different open source simulation software is assessed according to its modeling capacities for smart grids. First, a conceptual model for smart grids is formulated. The model is based on the idea of multiple flow networks and is used as a reference for the evaluation of open-source power system software tools. Further, the section includes a description of the different simulators and a power flow simulation test case with the IEEE 14-bus system.

Chapter 4 is dedicated to the open source Renewable and Alternative Power System Simulation tool RAPSim. It has been developed by the author and several students over the past years. Chapter 4 starts with a list of requirements that state the goals of the development process. A general description and explanation of the main design principles is followed by an example of model implementation and a test case. The chapter's conclusion discusses how well the requirements were fulfilled and what further steps should follow.

Although the previous two chapters are closely related, chapter 5 is a specific example of what the combination of ICT and power engineering can mean. The investigation of power profiles, as used in load disaggregation, by an information theory approach leads to the definition of proficiency. Further, the capability of proficiency as a measure for the difficulty of load disaggregation problems is tested on data based on measurements. Chapter 6 contains a conclusion with respect to the problems stated in section 1.3.

CHAPTER 2 Smart Grid Models and Components

"The whole is greater than the sum of its parts."

– Aristotle

Smart Grid as a technical concept stemmed from the idea of using ICT for improvement in electric power grids. The development of ICT and the need for integration of renewable power generation into the grid are two reasons that consequently led to the development of the Smart Grid. In that sense, two different technological fields merge, which leads to many new possibilities, as well as challenges. Although there is a concrete overall goal, the way to achieve it, technical capacity to achieve it adn what is economically favorable requires much discussion and research. These discussions are hindered by the differences between the involved disciplines, including power engineering, computer science, business administration and many others.

2.1 Perspectives on the Smart Microgrid

The following two sections summarize two different views of the smart grid. The first is a very general architectural model with the aim of defining terms and rough operational areas to achieve interoperability on all levels, from the power grid up to business processes. The second view is focused on engineering where power systems and ICT amalgamate and form a cyber-physical system.

2.1.1 The Smart Grid Architectural Model

The Smart Grid Architectural Model (SGAM) [Tre12, CEN12] was initiated by the Smart Grids Coordination Group of the European Committee for Electrotechnical Standardization CEN-CENELEC-ETSI to provide a reference model for smart grid architectures. It

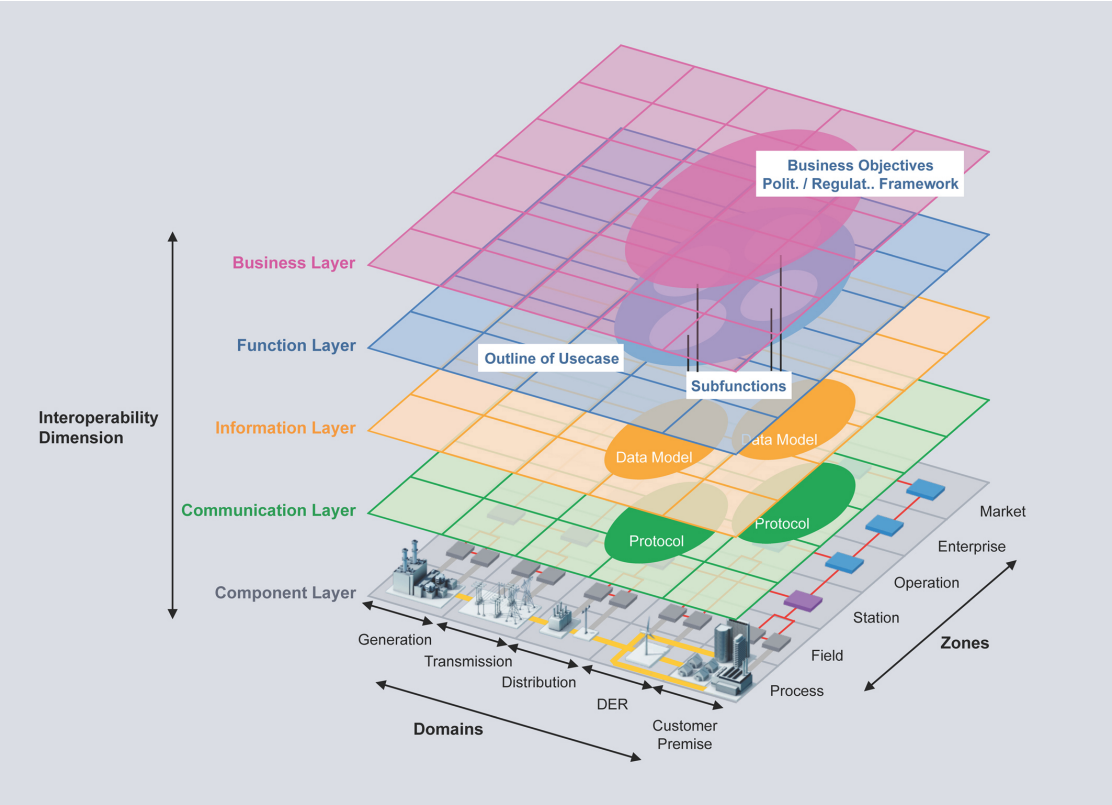

Figure 2.1: The SGAM describes three dimensions [CEN12].

was first published in 2012 and is intended to aid any technical and commercial cooperation. It divides the SG-domain into cells along the three dimensions interoperability, hierarchical process zones and the physical electricity domains. The SGAM is useful to analyze use cases, identify involved cells, and, for example specify required interfaces within the total system. In [Bod15] the usability of SGAM for MGs was evaluated. Within the concluding remarks is stated that for MGs the same conceptual model should be used as generally for smart grids, even though the focus might be different.

The five domains of the SGAM refer to zones in the power system, i. e., generation, transmission, distribution, DER and costumer or premise. The process zone contains the primary technical components of the power system. The other zones aggregate those parts. The Field zone contains the controllers to keep the power balance. The superior Station zone aggregates field components, which contain processes. Those components are intelligent, can communicate, and can send and receive commands. The Operation zone contains the grid stability systems or local energy management systems. Enterprise level contains business processes or offered services. Finally, the market zone describes the business field or type of energy market. The interoperability dimension of the SGAM describes five layers. The component layer is the physical device which is smart, i. e., able to communicate on the respective layer. The communication layer can be split in a physical and a logical layer that must fit each other. All the

protocols to enable communication are refer to that layer which can be further divided according to communication reference models, like the OSI-, or DoD layer model. The information layer contains data models and information objects which form the semantics for interoperability. The documentation and organization of functional data is described on a more or less general level. Finally, the function layer describes functions and services that are provided within a smart grid independent of their implementation. Interoperability with SCADA functions for control, notification and others is required. The top layer describes the exchange of information from the business administrative perspective. It can help to develop new business models as stated in [Bod15].

A feature that distinguishes this European reference model from, for example its US counterpart, is the existence of a separate domain for DER. This is consistent with the popularity of, roughly speaking homemade electricity especially in Germany and other EU-countries. The DER domain was created with the inclusion of, e. g., rooftop PV systems, MG, virtual power plants and similar concepts into the overall meshed power grid on the distribution level in mind. In addition to the SGAM, there exist other attempts to define reference models for smart grids, like in [Ces13] or in the RASSA[1] project (with a specific focus on security issues). Even though it is possible that SGAM will be supplemented by other ideas it is a quite comprehensive reference system. It is meant to enable interoperability of future and current smart grid technology at all levels from single components up to business processes.

2.1.2 Cyber-Physical Energy Systems

A cyber-physical system results when a physical system is interconnected with an Information Technology (IT) and communication system. Even though the automation of the power grid has been ongoing for some years, a new level of IT penetration rises with the transition to the smart grid. The ubiquitous inclusion of data and information of any kind brings new opportunities but also well challenges. In simulations, two different paradigms clash against each other: Physical systems as the power grids are commonly simulated by models based on continuous differential equations. While the IT processes and communication protocols are simulated discretely and in an event-based manner. The combination is generally not simple and has been the subject of research for some years [Ili08, Wid12]. The existence of this problem, sparked the development of suitable simulation frameworks [Pal14, Wij13, Far12].

2.2 Components in Active Distribution Networks

As mentioned in [Cho09, 2] traditional distribution systems have developed from a power-consuming black-box towards an active distribution network with bidirectional

[1] http://its.fh-salzburg.ac.at/forschung/forschungsprojekte/uebersicht/details/article/rassa/

power flow. Consequently the distribution system can support superior grid-systems in power supply and grid stability. Many different actors, in a technical as well as in an economic sense, are involved in different functions within the smart grid. A MG itself consists of a couple of these actors and simultaneously can be an actor in a superior smart grid. The smart grid actors presented within this section focus on the consumer and prosumer perspective. Specific properties and implications for their simulation in software are highlighted.

2.2.1 Renewable Energy Sources

Renewable generation is certainly becoming ever more popular. The deployment of renewable sources gained momentum from the evidence that fossil fuel-based power generation drives global warming. Aside from environmental compatibility, their popularity is driven by other factors such as the almost negligible production costs for power after the initial investment, governmental subsidies (provided in many western countries), the ability to supply power without the grid infrastructure, and the possibility of self-sufficiency of private properties. The number of renewable sources is increasing, this is true for the number of relatively small islanded facilities in remote places, as well as grid-connected versions of various sizes. Even though the requirements on controlling are different in these two operation modes both must deal with the fluctuations inherent to renewable power generation. Costs of renewable production are widely independent of the load factor. Consequently, the cost of power production is almost negligible but the power is not unlimited or in the desired quantity. Therefore controllers aim for the best power utilization and any losses are avoided. Alongside other functions [Cho09, p61], the source controller aims for voltage stability.

The traditional grid control paradigm is that production follows demand. That strategy is based on more or less adjustable generation facilities. Even though it makes stability control comparably simple it is not possible without careful planning. The expected demand is estimated ahead of time to get production facilities ready for delivering at the scheduled time. A reliable estimate for the demand is therefore necessary and is accomplished statistically, by aggregating many individual (residential) consumers. Unpredictable and random load events cancel out somwhat to yield a relatively predictable total load profile. The energy production to feed this profile was selected by scheduling the generators under economic constraints.

Better control through more information

Individual households' consumption patterns are not perfectly synchronized due to residents' different habits and the distinguishing appliances in use. Even though most people turn on the light when it gets dark or start cooking around noon, they do not simultaneously push the button when the second hand hits twelve. On the other hand, clouds covering the sky affect all PV- systems in an area simultaneously. Wind turbines within a single wind farm are all largely exposed to the same driving force. Reduction of the impact of fluctuations in renewable production requires enhanced control strategies

as offered by smart technologies. In general, the goal is to have more information about expected production early enough to settle on control actions. This is enabled by sensor penetration on all levels, data integration of other fields, ubiquitous computing and other arrangements. One example is the prediction of PV output using images from weather satellites. This example makes it obvious that other renewable sources require different prediction methods, i. e., each type of production requires different techniques.

Data about the expected outcome are required in both operational control and in the initial designing phase also. Decisions such as component sizing or installation site selection of can be made on assumptions about the outcome. If the data about sun radiation intensity are available, they can be used to make reasonable prognoses about PV production for specific scenarios. Of course, the PV production characteristics must be well-known, seasonal climate changes must be considered, and local conditions are not identical for the same time period in any given year (e. g., the same month of the subsequent year might not be comparable). Measured data from comparably facilities are a valuable source for the design of renewable power production plants. Different models and methods have developed for the different types of renewable generation to assess expected outcomes.

PV generation is limited by the total sun radiation intensity on the Earth's surface. The layout of the solar system is well-known and the radiation in the outer layers of atmosphere can be calculated precisely in advance for years. After taking into account scattering and other effects the clear sky radiation gives an upper bound for the maximum possible power produced by a PV system. Such a bound does not exist for wind generation. Wind speed can reach values such that turbines must be stopped to avoid damage or even destruction. Therefore, time series of wind generation have totally different characteristics than PV production, which shows characteristic periodicity with daylight. Tidal hydro power generation has also periodic and well-predictable behavior [Zha14]. Conventional hydro production with a reservoir enables at least some control for renewable generation.

Different types of renewable power generation have very specific power profiles that further depend on the specific device and its design. The knowledge or simulation of characteristic and realistic power profiles is important to reliably design renewable production facilities, regardless of their size. Forecasting models, which are essential for efficient control, are developed and tested based on measured data. The characteristics of power generation profiles depend highly on the type of renewable source. The different technologies have different constraints that affect modeling, control and facility design. Therefore time-series, regardless whether they are measured data or artificially generated, are required to model the expected values for power delivery.

2.2.2 Residential Demand

It is important to know the residential power demand in large-scale distribution networks. Grid providers have learned to estimate the expected load by considering weekly and

seasonal variations and even single, predictable events. It is known that averaged load curves depend on the day of the week and the time of day. For the case of a single or several aggregated household it is much harder to get reliable demand profile predictions. Single household power profiles are derived either from measurements or statistical models. Several studies include measurement campaigns [Mon14, Kol11, Bec14] which are done with one single point of measurement or even on device level with different sample rates. Other studies [Got11, Arm09] yield device level models which are normalized by averaged data. Such models are used to simulate single households and predict expected consequences of load shifting activities. Methods such as Demand Response (DR) shift residential consumption from a totally passive to an active component of the distribution system. Whether DR systems will become widespread in the distribution system is unclear as yet. The difficulty is not so much the technical feasibility as the related investment costs and the lack of refunding for dropped off power in the current power system business. Therefore, the new role of aggregation in power systems could be a possible solution. As residential demand represents roughly a third of total demand it is important to find realistic load shifting and power saving methods to reduce carbon dioxide emission.

To make consumers contribute towards that goals, a local energy management system is requird, with the ability to control shiftable loads and, optionally, generation facilities. At the same time such systems must be low-cost as potential savings differ vastly between appliance configurations and are generally low, as shown in [Got11]. To equip each device in a household with a sensor greatly increases cost and therefore techniques with a single point of measurement are beneficial to obtain detailed devices status information. Load disaggregation, also called NILM (introduced in section 1.2 and 5.1), is such a single device measurement technique. Just as for residential load modeling NILM requires knowledge about single device usage, either during development or for AI learning phases. Single residential consumer profiles are hard to simulate as they are correlated with behavioral patterns of the inhabitants. Those patterns show regularities for wide time windows but irregularities and singular events are very frequent. The ability to model single residential load profiles is currently more interesting for development of future technologies, such as DR, than for grid controlling.

2.2.3 The Microgrid

The MG [Las01, Las07, Cho09] is a concept where one or more distributed energy resources are combined with local consumption into a single entity. The MG can be operated in island mode or be connected to a superior distribution system, which means it either feeds into the grid or takes power to feed local demand. MGs have historical applications in the supply of islands. That includes both infrastructure (such as alpine cottages isolated from the main grid) and geographical islands. Mariam et al. show examples for MG architectures on islands in [Mar13] which includes Bornholm (Denmark), Kythnos (Greece), and others. A MG can be operated as a DC-system, like

for remote houses or onboard vehicles, as buses, cruise-ships etc. DC is favored when the system a relatively low power one, a battery is included and generation is also in DC, like a PV-system or a fuel cell. AC MG systems are preferred to supply geographical islands [Lid11] to ensure interoperability of user devices. Almost all backup generation sets with combustion engines produce AC. They are used to supply remote areas or as an emergency supply for critical infrastructure like hospitals or military facilities. Those traditional MGs run exclusively in standalone mode. The MG, as an actor in the smart grid, can connect to the superior distribution system or decide to run on its own. The grid connection obviously helps the MG to saturate higher demand. In return, the distribution grid can benefit from the MG due to its support for power quality. Additional benefits of MGs are reduced overall losses by waste heat utilization, released capacity and fewer losses on the transmission system.

A central MG controller is required to provide power of sufficient quality in island mode and manage coupling to the superior grid. The controller regulates a sufficiently powerful source to achieve load balance and voltage stability. This is usually a diesel generator, but a Combined Heat and Power (CHP) sources or a storage system could be alternatives. The MG is then termed demand-driven as the current production is adjusted to fit the consumption. International transmission grids are traditionally controlled in this way, with the major difference that generation is scheduled one day ahead to fit the expected consumption. Such early planning is required because many power generation facilities have enormous boosting times. In a MG only some generators are controllable, whereas renewable sources, e.g. PV and wind generators, are widely uncontrollable in their power output. The inclusion of renewable power sources into a MG is of course highly desirable. Aside from their ecological friendliness they do not require any combustible material and provide power that is almost free, except for the initial investment. The drawback is that they cannot be arbitrarily scheduled and are difficult to control. A reduction of output is possible but is equivalent to losing usable energy, which is undesirable and is only justifiable to prevent damage. Therefore, integration of renewable sources requires additional effort in the control system of MGs [Col09]. To maintain the power balance and also compensate fluctuations in production requires flexibility of power, in generation and/or consumption. This power flexibility in a MG can be either provided by a controllable generator, by controllable loads, by storage or by connection to the distribution-grid.

Time Series in Design

A storage system can act as a consumption or a generation entity (however needed). The most important parameter of a storage system is its capacity. The determination of the proper storage size for a specific MG requires knowledge about its characteristic production and consumption profiles. To avoid shortages, the average production must exceed the average demand (in the long-term there cannot be more power consumed than produced). The production-consumption mismatch can be compensated for a specific time, which is a main parameter of the storage system. That means basically that a storage system to balance a MG between day and night can be smaller compared to one that

balances a summer winter disequilibrium. Obviously the storage loss, maximum input and output power and further parameters must be considered in storage design. However, storage is expensive and consequently other ways to gain controllability are used also. Additionally, MGs can benefit from reliable preplanning, just as transmission systems do. That requires available forecasts for fluctuating components such as renewable production and especially user-driven consumption. Also, long-term planning allows utilization of sources with higher ”ready to work” time for balancing. Nevertheless, to control a MG efficiently requires the knowledge of characteristic time series for consumption and production. A widely used practice is to combine measurements with approximations and add a safety margin to achieve a certain security of supply, just as transmission and distribution grid operators gained experience in forecasting the demand for specific cities or companies. The challenge, especially for small MGs is their medium to small size - they are too large to control every device individually and too small to be treated by statistically averaged values, i.e., the law of big numbers does not apply. Possible solutions can come from AI supported systems. They are based on data collections (over a for long period and for specific places) that allow projections of expectations and the integration of other data sources, which could be weather forecasts, smart household devices, and residents' mobile devices. However, for a solution to be designed requires some amount of reliable data.

CHAPTER

3 Survey on Smart Grid Simulation

"There is something irreversible about acquiring knowledge; and the simulation of the search for it differs in a most profound way from the reality."

– J. Robert Oppenheimer

This chapter gives an overview what is currently cutting edge in smart grid simulation. It provides an introduction and an evaluation of open source smart grid simulation software and a survey on software for renewable energy source simulation, especially Photovoltaic. First, a generic analytical model is defined for smart grids on the concept of flow networks, with the aim of obtaining a more complete model of the processes in smart grid control and development. This generic model is further used to compare ten different simulation tools according to their feasibility to model the smart grid more complete. This is followed by a brief description of the simulation tools. Subsequently, generic descriptions of four simulators are tested by applying them to a reference test case of power flow studies. The results are presented in detail, including a user report for the four selected simulators. A short version of the work was first published in [Pöc13]. Within section 3.3, there is a summary of current renewable power source simulators This content was created by a co-author of [Pöc14].

3.1 Layers of Flow Networks

The traditional grid is characterized by a unique power flow direction and a steady regulation scheme based on voltage level and frequency. Energy production is regulated to satisfy the current overall consumption and assure grid stability. The power flow within a smart grid is likely to be more complex as production also takes place in the distribution system and many renewable sources do not generate power as consistently as fossil-fuel powered sources. To produce grids of greater efficiency, reliability, and environmental sustainability without compromising on stability requires more sophisticated control concepts. This requires the contribution of all players in a complex smart grid system

and can no longer be performed by grid operators with simple production scheduling. Weather and meteorological conditions can cause overproduction at one time or high energy demand at another. Other variability is related to residential load profiles. User behavior and related power consumption depends on cultural habits and social norms. For instance a consumption peak my relate to sport broadcasts or late breakfast at weekends. More and better distributed information that arise from the ”smartness” of the grid is a potential solution for dealing with the related uncertainties. To realize much of that potential requires better data processing, real time grid state monitoring and demand and production prognoses. To increase the number of contributors for energy supply and grid stability needs easily accessible markets for different power services. The operators of small scale power facilities will find ways to participate at existing markets for day-ahead power, control energy, and also carbon dioxide emission certificates. Even if real time pricing is not expected in the near future, any power consumption (as well as production) is associated with a price that consumers pay to providers. The expected monetary gains will attract power traders as well as consumers and will drive investments in R&D and infrastructure. Some investors might be motivated by environmental sustainability, renewable energy usage, and emission reductions over financial gain. However, almost all related activities happen in an economic context. Therefore, the payments flow is a clearly defined way to track activities and dependencies between power grid agents.

3.1.1 Motivation

The electric power system is a network of loads and feeders connected by power lines, just as a complex network consists of a set of nodes interconnected by edges (= links) [New10]. Networks are established tools in complex system modeling. Any flow is a directed link connecting its source node with the receiver node. The flow intensity gives the weight of a link. Flow is a physical property that in general can be measured, which could allow real world data to be used for modeling. A smart power grid model becomes more complete by including two essential components: the flow of data in the ICT network and the flow of payment through the financial transaction network. For agent-based models it is proposed that the flow of information and the flow of payment should be considered in addition to the flow of energy. This is for the following reasons: (i) Energy, payment, and information (at least by amount of data) are measurable dimensions. (ii) All three types of flow are essential for the control and the development of a smart grid system. (iii) The concept does not need further assumptions and is unrestricted by technical circumstances.

A simple example of a smart grid scenario includes a user, a grid operator, an electric vehicle (EV), a switching device and a smart meter. Electrical power is transmitted from the smart meter (energized by the grid), through the switch into the storage of the EV. Information is transmitted from the smart meter to the user and to the grid operator. The EV itself or the user informs the switch to connect or disconnect the EV. Financial transactions take place between the user and the grid operator. This simple example

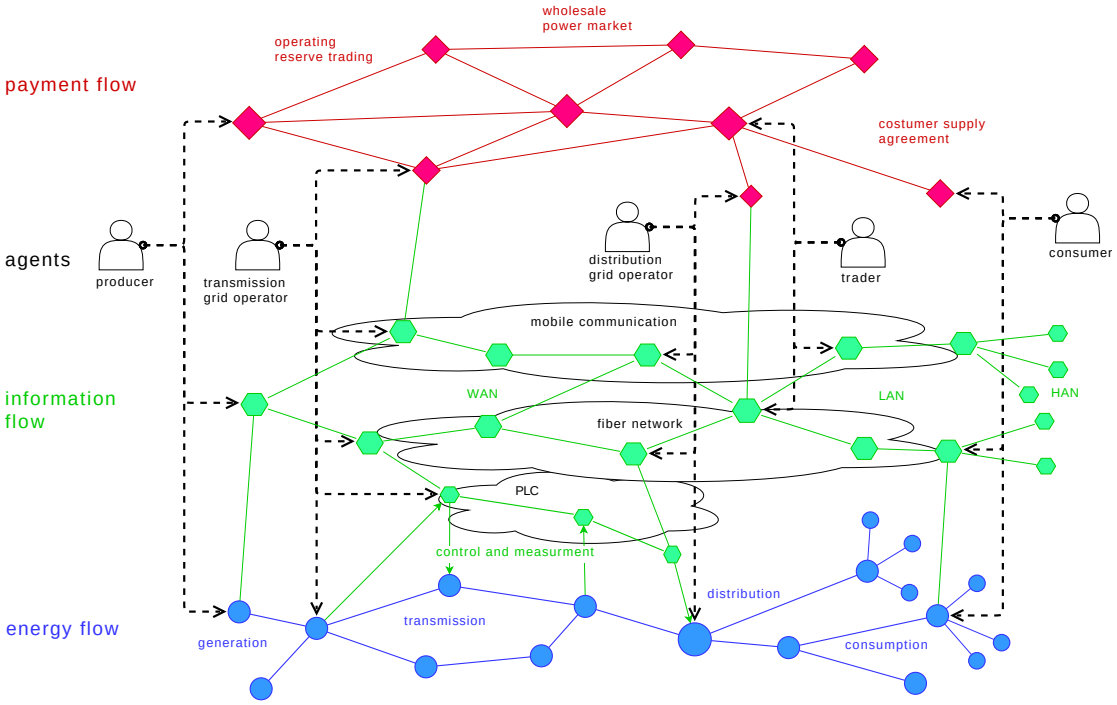

Figure 3.1: A complex smart grid system modeled with three networks of flow (energy, information and payment) and several agents. The nodes (circle for energy, hexagon for information, diamond for payment) in the network modify the flow, i. e., they are source and/or receiver of it. Nodes can have very different functions with respect to their location, e. g., power generation, LAN routing, or power market operators. The agents are the decision-making entities and involve several nodes and are the main interconnections between different flow networks.

shows that the three types of flow networks are tightly interwoven. A single device can be a node in multiple flow networks. For instance, the smart meter is a source of both power and information. The person, in this example controls the car, can unplug it and pays for the energy. The person is an agent, a decision-making entity in the system that tries to optimize its utility function by controlling several devices which are nodes in the flow networks. Within the complex smart grid system an agent is a type of super node that pools several nodes from different flow networks. Every device or instance that modifies flow can be seen as a node.

Figure 3.1 shows a larger example of flow networks involving five agents that exchange the three different types of flow. Blue links indicate flow of energy, green links the information flow, and red links, payment. The three networks are shown with different colors and node-symbols: blue circles for energy flow nodes, green hexagons for information flow nodes and red diamonds for nodes in payment flow. The dashed lines connect an agent with its controlled nodes. Those nodes are associated in the sense that they are part of the same agent, but do not need to be connected by a direct link of flow.

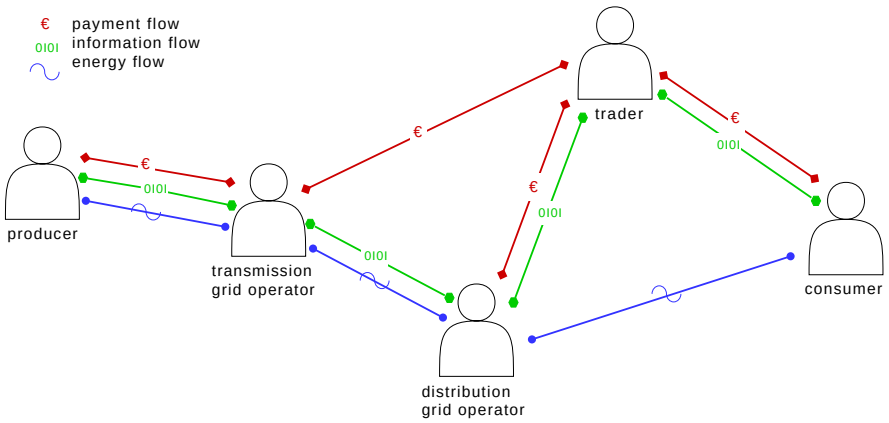

Figure 3.2: Several smart grid system agents interact in terms of energyflow, informationflow and paymentflow, forming a complex network with multiple types of links. This is a reduced version of figure 3.1.

The agents are an important interconnection between the different flow networks. All the nodes, even when not assigned to an agent, modify flow according to their function in one or more flow networks. Measurement and control processes, for instance, are frequent links of information flow generated or consumed by typical energy nodes. The division line between the layers of flow should be kept clear. A SCADA-system could equally be assigned to the power or the information flow system. The majority of links are expected between nodes of the same type as those are technically predefined. A simplified version of this system of flow networks is shown in figure 3.1.

3.1.2 The Components in the Flow Networks Model

A description of building blocks for the complex flow network model for smart grids is discussed in the following paragraphs.

The Agents

Smart grids can be modeled as multi-agent systems as in [Pip09, Häg10, Oli12] or for power market simulations. Smart grid agents can be assigned different roles e. g., consumers, producers, service providers, and mixed forms of each role, as a grid operator that owns production facilities. Each agent has a utility function which is designed according the main intention or the goal the agent strives for. The agent optimizes its utility function within its available scope of actions. It aims to balance the three flow types according to the opportunities available to it. In this setting the agent can act on a given flow type only through the nodes it controls. Any social relation among agents is not captured explicitly but may be reflected in information flow. In the set of flow

networks, agents relate different node types to each other. Building a specific model for an agent is a rather challenging task.

Energy Flow in the Power Grid

The power system is the basic layer of the complex flow network model. It consists of power generating nodes, consuming nodes, and transmission nodes. Nodes can be both generation and consumption nodes at different time, such as storage devices and Electric Vehicle (EV)]s. Transmission losses need to be included additionally as power sink (equal to a source of negative flow). In that sense, non-source nodes are not explicit generators or consumers. Reactive power can be captured by the model as well even though its influence is way below that of active power. Three basic principles for a flow f from a node A to a node B could be used as in current power system analysis methods. These are:

- Antisymmetry of flow: $f_P(A, B) = -f_P(B, A)$;
- Limitation of transmission capacities: $f_P(A, B) \leq c_P(A, B)$;
- Equality of flow for non-source nodes: $\sum_B f_P(A, B) = 0$.

Any general function f_P can be used to describe the flow of power P (denoted by the subscript). The unit for measurement is kWh per time step, which is technically an average power. The model-specific time step depends on the process of interest. The time steps could be days, as in case of wholesale markets, or seconds, as in grid stability control. The power grid has an extensive control system to run it safely and reliably, e. g., SCADA, which can be seen as part of the power flow or information flow as well.

Information Flow over the ICT Infrastructure

The information flow network is in some aspects special due to its numerous substructures and the wide meaning of information. In a smart grid system there are very different types of information, which are measured data signals automatically processed for device control or security purposes. There is a lot of critical information, in terms of pricing, privacy or device security issues. Further, large amounts of data are related to energy markets, e. g., the bid submission that mostly runs online. Information flow is based on the related data transmission. Technically it is the data flow which can be measured in Bytes per time step. There is a difference between the data transmitted between two nodes and the actual information gain for the receiver. The receiver requires additional knowledge for data interpretation and known facts do not represent a gain of information while still producing data traffic, but this is inherent to the concept of information. In contrast to power, data can be stored and transmitted very easily, although the capacity for this is not unlimited, with transmission rates being limited by the capacity of the

transmission channel and data storage being limited by the memory of the nodes. For data flow applies the

- Limitation of transmission capacity $f_c(A, B) \leq c_c(A, B)$

but not antisymmetry and flow equality.

The whole data flow is based upon different technologies with different characteristics. Data receiving nodes are usually addressable and embedded into a network infrastructure. All the available ICT infrastructure can be used for smart grid purposes. The technologies differ in time constraints, data rates, security standards, and whether they are single-, multi-, or broadcast communication. It can be helpful to constrain the model to a subsystem of the information flow. That can be defined, e. g., according the used OSI layers, type of transmission media, or used protocol standards. Most of the established ICT infrastructure is used for smart grid purposes [Gün11, Fan11]. It is possible to split the total information flow network into different communication subnetworks, which could be distinguished by their physical transmission media, e. g., wireless, fiber, or power line communication, or by the applied communication protocols (e. g., TCP/IP, ZigBee). Alternatively, the differentiation of data flow can be done by its origin or application area, e. g., user data, market-related, control and measurement, SCADA system, or environmental data. An agent will generally try to collect useful information and keep other information as private as possible. Physically the whole data flow depends on the availability of electrical power.

Payment Flow at the Finance Transaction Network

A node in the finance transaction network is somehow an account. The owner of the account takes a role in the energy market. They set bid offers to sell energy or buy energy from the market. Less obvious roles of nodes in the payment flow network are market operators and service providers. The operators of energy infrastructure for instance offer delivery services and pay other companies to construct or maintain power lines. It is expected that with further development of the smart grid the market for energy related services will enlarge, perhaps through metering, price and demand prediction, management of EV charging, etc. The flow, in accounting, arises from the fact that a sum appears at the receiver which left the sender shortly before. A transaction to any other account around the globe requires only the receiver's number. An account itself accumulates the payment flow over time. It has no maximal, only a minimal state that can be negative and limits the maximal outgoing payment. The state of the account is a major part of the utility function of the superior agent. For payment flow applies

- antisymmetry: $f_f(A, B) = -f_f(A, B)$.

In general, there are no losses in financial transactions but all additional costs, e. g., for accounting services, should be regarded as losses. The general rules behind the finance

transactions originate from bilateral contracts or energy markets embedded in a legislative system. An energy market is a hub in the finance network, as all participants trade through the market instead of bilateral contracts. The payment flow network should capture dependencies between smart grid agents. Financial transactions are no longer based on a physical network as there are no transmitted media. The payment flow layer is situated above the communication flow layer in the sense that financial transactions require safe ICT connections outside the smart grid area.

3.1.3 Summary

The complex smart grid flow network system is a generic model. It helps to structure and classify smart grid problems and scenarios. The model can be scaled to any type of problem or subsystem, from intercontinental transmission lines to household devices. A node can be the size of a power plant, a single device, or even a small part of it. The model can be extended to other flow networks, e. g., the gas distribution system and other forms of energy, or by a network of the agents' social relations.

Flow is a measurable quantity, which is an important requirement for extending the model quantitatively and for evaluation against real world data, which requires a description of the nodes' functions and might enable application of flow network mathematics. With the exception of the power system, the model is currently far from quantitative simulation. The model can be related to other general models, as now demonstrated for the three dimensions of the SGAM. Power flows in the components layer, along all the domains and mainly in the zones between process and station. The information flow is tied to all domains and zones of the communication and the information layer. The payment flow can be assigned to the business layer at all domains in the zones from operation to market. The principal difference is that the flow network model indicates interdependencies of agents, whereas the SGAM aims to structure the smart grid space and enable interoperability.

3.2 Software for Power System Simulation

As in many other fields, simulation is a valuable tool for performing virtual experiments for exploring research and design questions. Simulation requires, first of all, a computational model of the system under investigation which can be evolved over time. The rules or equations for the time evolution are part of the model. Most important is the appropriateness of a model for answering a specific question. Currently there exists no general smart grid model or simulation approach. Conceptual models, like SGAM, are helpful to structure different types of problems but are less suitable for technical simulation. Simulation of conventional power systems has developed substantially over the past few decades and is mainly utilized for planning and operation of transmission grid systems. As the backbone of the power system, they are critical and outage must

be avoided. Traditional distribution systems are mostly passive which facilitates their smooth operation. Distribution systems are monitored less and face less difficult control tasks than transmission grids. Consequently, distribution system simulation has not been a burning issue so far. With the popularity of DER increasing, like PV or CHP plants, and active consumers, with DR and smart EV charging stations, this will change.

3.2.1 Physical Power Grid Simulation

Calculating transient functions of electromagnetic fluxes has been an issue since rotating generators were introduced and since AC power systems have been in place, so it is no wonder that early computers were utilized to solve the related differential equations. Electrodynamic simulations have advanced since then and are widely used for power system design and analysis. Many different professional software tools as well as open software are available[Mah09]. The simulation of electricity systems is a traditional field in engineering that distinguishes between dynamic and static power system analysis. Current software tools are simulators that perform various analyses within that field as well as in related fields. There is a broad range of applications in industry and science and commercial software developers provide a variety of software packages, e. g., IncSys[1], Pacific Northwest National Laboratory[2], NEPLAN[3], GE Energy[4] or PowerWorld Simulator[5]. However, these packages are often costly, highly specialized and hardly modifiable, which means they are poorly suited research and/or teaching. The IEEE Task Force on Open Source software for Power Systems aims to address this shortcoming and has published a list of freely available software tools for power system simulations online. The tools are compared in table 3.1 and are briefly described in section 3.2.4. Four selected simulators are additionally applied to a test case within section 3.4.

The aim of transient simulation is to model the physical layer in as much detail as is required (although making a model more detailed is uneconomical). In terms of safe and secure operations and stability the accurate simulation of power systems is essential. The physics of the power grid is a precondition that any smart grid use-case has to respect. However, many scenarios happen on timescales much slower than the electrodynamic transients. For these slow processes the grid appears to change its state instantaneously. The different simulation areas distinguish themselves significantly by the different timescales of the modeled phenomena as shown in figure 1.1.

[1] http://www.incsys.com/powersimulator.htm
[2] http://availabletechnologies.pnnl.gov/technology.asp?id=288
[3] http://www.neplan.ch
[4] http://www.ge-energy.com/products_and_services/products/concorda_software_suite/index.jsp
[5] http://www.powerworld.com/products/simulator.asp

3.2.2 Market Simulations

The electricity market has a major influence on the transmission grids of the respective regulation zone. The electricity supply business is undergoing a dramatic change due to the liberalization of consumer markets. According to the new perspectives of smart grid the market is expected to regulate the electricity system even more than now. Market-based controls are expected to decrease CO2 emissions and are also used for home energy management [Mon16]. A cascade of markets that trade energy at different timescales enables grid balancing, which is mainly the day ahead market with twelve to thirty-six hour delay between an auction and power delivery combined with the trade of control energy at reaction times of minutes and even seconds.

The smart grid's development is leading to increased density of the connections of the electricity system to business processes. SGAM is helping to structure the possible connections of electricity facilities with markets or other businesses. Consequently, it is no surprise that market simulations play a role in recent energy systems research. In addition to other approaches [Bom08], marked simulations are often performed using agent-based models [Vyt10, Pip09, Con05]. A publicly available software tool was created with AMES [Li09b, Li09a]. A detailed joint simulation of the physical power system and the market processes would be even more technically demanding, and its necessity and applicability are as yet unclear.

3.2.3 Comparison of Open Source Power System Simulation

The power system has a key role within the complex smart grid system as the infrastructure that enables any flow and trade of power, and the change in this system defines the state of the grid. This is the motivation for taking several power system simulators and assess their potential capabilities towards more comprehensive complex smart grid simulation.

Table 3.1 contains a list of functionalities provided by ten different power system simulators. Their major features are listed and they are assigned to four different areas related to complex flow networks. The areas and functionalities are explained in table 3.2. The tables do not contain a DC-PF function as the simulators can either solve for AC-PF or optimal DC-PF. DC-OPF is commonly used for economic analysis to optimize welfare and benefit in accordance with a power system. Owing to its ease of solving, it is used more frequently than the AC version of OPF [Pur05]. All the power flow analyses are static or quasi-static, e. g., in AMES where twenty-four values are used in one day . Within SA, several forms of stability analysis are considered, voltage or small signal stability as well as outage scenarios or short-circuit simulation. All of the energy market simulations provide different options for market rules which are not listed as separate functions in table 3.1, e. g., for market clearing mode or bid and offer submission. Most market simulations use ABM, which is considered as a separate feature for complex systems modeling.

Software	Power Flow Functions					Power Dynamics			Information		Market		Specials
	AC-PF	DC-OPF	OPF	CPF	3P-PF	EMT	SA	TD	PLC	COM	MKT	DR	
UWPFLOW	✓	✓					✓						
TEFTS						✓		✓					
MatPower	✓	✓	✓	*							*		GUI, GNE
PSAT	✓	✓	✓	✓			✓	✓					
IPSYS	✓	✓	✓										GNE
MatDyn						✓	✓	✓					
AMES		✓									✓	✓	GUI, ABM
InterPSS	✓	✓			✓	✓	✓		✓		✓		GUI, GNE
OpenDSS	✓				✓	✓		✓	✓	✓			GIC
GridLAB-D	✓				✓		✓	✓	✓	✓	✓	✓	ABM, Climate Data

∗an optional package is available

Table 3.1: Functionalities of ten free available power system simulators.

The first two and oldest simulators in table 3.1 are designed for static or dynamic power system analysis, respectively. The four simulators in the second group, from MatPower to MatDyn, run in the Mathworks MatLab environment, which as commercial software became popular through the huge variety of available toolboxes. The last four simulators in the list go beyond pure power system simulation. AMES is explicitly designed for energy markets, whereas the others simulate power systems and have optional market - related functions.

Table 3.1 highlights that more recent simulators have a greater range of capabilities than were offered previously, and this is due to the increased number of simulation tasks arising from the development of smart grids. Distribution systems have been captured in more detail by recent simulators [Sch09]. The modeling paradigm changes and well-defined differential equation models are enriched, for example, by agent-based

Power Flow Functions	
AC-PF	AC power flow
DC-OPF	DC optimal power flow
OPF	optimal power flow, the general AC version
CPF	continuous power flow
3P-PF	three phase power flow for distribution grid
Power Dynamics	
EMT	electro magnetic transients
SA	stability analyses, various types of
TD	time domain simulation
Information Flow	
PLC	programmable logic controllers
COM	communication links
Energy Market	
MKT	energy market simulations
DR	demand response, by consumers or producers
General and Special Functions	
ABM	agent based model
GUI	graphical user interface
GNE	graphical network editor
GIC	geomagnetically induced current
	climate data integration

Table 3.2: Functionalities of the power system simulators with the abbreviations. They are listed and grouped as in table 3.1.

modeling, stochastic models or game theoretic approaches [Bom05]. The simulators interconnect to other disciplines, primarily to economics but also to ICT systems [Nut07]. The implementation of climate data, as in GridLAB-D, or other forms of energy[6] are further examples therefore. The code from newer simulation software is more modularly structured and integrable into other programs, which makes it easier to extend and adapt programs according to the requirements of a specific research problem. In some cases researchers still design their own software, especially for development of alternative

[6]IPSYS (same name as in table 3.1 but different software) is a flexible simulation framework for integrated energy systems developed at the Technical University of Denmark [Bin04]

methods or for very new and specific problems. The Mosaik [Sch11, Sch12] simulation framework[7] is an example wherein existing simulators were recombined to form a new simulation framework. Researchers from different fields can perform simulations with the help of Mosaik's easily comprehensible Graphical User Interface (GUI).

3.2.4 Description of Open Source Power System Simulation

This section gives a brief description for each of the simulators in table 3.1 with respect to opportunities for integration into a more extensive smart grid simulation system.

UWPFLOW [8] (first published 1996, latest update 2010). This is a research tool that has been designed to calculate local bifurcations related to system limits or singularities in the system's Jacobian. The program also generates a series of output files that allow further analyses, such as tangent vectors, left and right eigenvectors at a singular bifurcation point, Jacobians, power flow solutions at different loading levels, voltage stability indices, etc. The program is written in C, and is available for Windows and UNIX, including examples and a brief tutorial. The tool is maintained by Prof. C. A. Cañizares, University of Waterloo, Ontario, Canada. UWPFLOW is a proven tool for several power analyses and is integrable into SG simulations.

TEFTS [9] (first release 1996). This is a simple transient stability program to research transient energy functions (TEFs) and voltage stability issues in dynamic models of AC-HVDC systems. The program is written in C, and two versions for DOS (16-bit) and UNIX are available, including examples and a brief tutorial. The application was hosted by the University of Waterloo, and although accessible in 2013, it was no longer available in 2016. The program completes UWPFLOW with its ability to analyze power system for short time intervals, i. e., high-resolution analysis. The missing PF functions make it less appropriate for complex smart grid simulation.

MATPOWER [10] (first published 1997, latest update 2016). MatPower is a package of MatLab M-files for solving power flow and optimal power flow problems. It is intended to be a simulation tool for researchers and educators that is easy to use and modify. MATPOWER is designed to give the best performance possible while keeping the code simple to understand and modify. It was initially developed as part of the PowerWeb[11] project. Although MatPower is free and open source, it

[7] http://mosaik.offis.de
[8] https://ece.uwaterloo.ca/~ccanizar/software/pflow.htm
[9] https://ece.uwaterloo.ca/~ccanizar/software/tefts.htm
[10] http://www.pserc.cornell.edu/MatPower
[11] http://www.pserc.cornell.edu:8082/powerweb

runs on MatLab, which requires a license. The application is developed by the Electrical and Computer Engineering group of Cornell University, Ithaca, New York. MatPower provides basic static power flow analysis. The MatLab environment allows many applications but limits the number of combinable systems.

PSAT [12] (first published 2002, latest update 2016). The Power System Analysis Toolbox is a GNU/Octave-based toolbox for electric power system analysis[Van07]. Further features of PSAT are: Phasor Measurement Unit (PMU) Placement; Eigenvalue Analysis; User Defined Models; Flexible Alternating Current Transmission System (FACTS) Models; Wind Turbine Models; Conversion of Data Files from several Formats; Export results to EPS, plain text, MS Excel and LaTeX files; and Interfaces to General Algebraic Modeling System (GAMS) and UWPFLOW Programs. PSAT was developed by Prof. Federico Milano at the University of Castilla-La Mancha, Spain. In 2011 he released a demo version of a new power system simulator called Dome which however is not open source. The developer has since left to work at University College Dublin. PSAT is one of the most complete power system simulators [Mil05] and very closely tied to MatLab. Integration into another system is faced with the same general difficulties as MatLab.

IPSYS [13] (first published 2004). The Interactive Power Systems Simulator is a scripting tool used to define, manipulate, and analyze electrical power systems data of a prescribed format. A user can interact with single or multiple power systems models through the IPSYS shell, a MatLab interface (MIPSYS), or a single network using a GUI Interface (GIPSYS). The IPSYS shell interface features can be grouped into four distinct types: data types, general operators and functions, program control and scripting, and specialized power systems routines. Power systems networks are modeled as separate objects that can be read in from a file, manipulated, modified, simulated, and optimized, and the resulting system can be stored in an external file. GIPSYS adds a GUI to IPSYS, from which a network can be entered, modified, and a few common algorithms executed. MIPSYS is the MatLab interface to IPSYS back-end routines. The software was developed at the Carnegie Mellon University [Ili07]. The interface coupling to MatLab and therefore MatPower is quite simple.

MATDYN [14] (first release 2009). MatDyn is a free MatLab-based open-source program to perform dynamic analyses of electric power systems. It was inspired by MatPower and shares its philosophy: it is intended as a simulation tool for researchers and educators that is easy to use and modify. The source code of MatDyn is available. Care has been taken to keep it well structured and easy to understand. The project is currently developed and maintained by Stijn Cole[Col11],

[12] http://http://faraday1.ucd.ie/psat.html
[13] http://www.ece.cmu.edu/~nsf-education/software.html
[14] http://www.esat.kuleuven.be/electa/teaching/matdyn

Katholieke Universiteit Leuven, in Belgium. The program completes MatPower with its dynamic analysis methods.

AMES [15] (first published 2007, latest release 2013). The AMES (Agent-based Modeling of Electricity Systems) Market Package is an extensible and modular agent-based framework for studying the dynamic efficiency and reliability of wholesale power markets restructured in accordance with guidelines issued by the U.S. Federal Energy Regulatory Commission. The AMES Market Package, developed entirely in Java by an interdisciplinary team of researchers at Iowa State University, is a free open-source tool designed for the study of small to medium-sized systems. AMES provides a graphical user interface and output reports through table and chart displays. The framework has some specialties and some weaknesses e. g., the hourly time steps might be to large for some applications, reactive power is not considered at all. On the other hand the learning tool for agents is an clear innovation. Modification requires some experience as a Java developer.

InterPSS [16] (first published 2006). The Internet Technology-based Open-source Power System Simulation System is an open-source development project aimed to develop an simple to use, yet powerful Internet technology based software system for design, analysis, and simulation of power systems. Its open and loosely coupled system architecture will allow components developed by others to be easily plugged into the system to augment its functionality, and equally importantly, allow its components to be integrated into other systems to provide certain power system simulation functionality or services. The project is currently under development by a team of developers living in the United States, Canada and China[Zho07]. Its open design and documentation makes it simple to understand and appropriate for complex system modeling.

OpenDSS [17] (first published 2009, latest release 2016). The Electric Power Research Institute, Inc. (EPRI) has made its Distribution System Simulator (DSS) program available as an open-source project on the sourceforge.net website. The OpenDSS software is a comprehensive electrical power system simulation tool for electric utility distribution systems. The program has been developed for the Microsoft Windows environment. It supports nearly all frequency domain (sinusoidal steady-state) analyses commonly performed on electric utility power distribution systems. In addition, it supports many new types of analyses that are designed to meet future needs related to grid modernization efforts. Additionally, this open source program may encourage the user community to contribute useful models that might help others active in grid modernization. It appears to be a powerful tool but takes some time for the user to familiarize themself with its operation. There are many specific models, such as those for advanced measuring infrastructure

[15] http://www2.econ.iastate.edu/tesfatsi/AMESMarketHome.htm
[16] http://community.interpss.org
[17] http://sourceforge.net/projects/electricdss

or distributed generation. A summary of key features, solution algorithms and example applications is given in [Dug11].

GRIDLAB-D [18] (first published 2010, latest release 2016). GridLAB-D is a flexible simulation environment [Cha08] that can be integrated with a variety of third-party data management and analysis tools[Sch09]. The core of GridLAB-D has an advanced algorithm that simultaneously coordinates the state of millions of independent devices, each of which is described by multiple differential equations. The advantages of this algorithm over traditional finite difference based simulators are: 1) it handles unusual situations much more accurately; 2) it handles widely disparate timescales, ranging from sub-seconds to many years; and 3) it is very easy to integrate with new modules and third-party systems. At its simplest, GridLAB-D examines in detail the interplay between every single different part of a distribution system. GridLAB-D does not require the use of reduced-order models for the aggregate behavior of consumer or electrical systems, which averts the danger of erroneous or misapplied assumptions. The import module for climate data is a unique feature. Further notable is the persistence of distributed modeling in the software. The scripting language is not very complicated but is specific to this software.

3.3 Software for Renewable Energy Sources

Renewable energy sources are developing in line with advancements in research. When PV systems run in island mode, e. g., in combination with a diesel generator, the sizing of system components and expected PV output are the most relevant factors to consider. Several studies and software tools address this issue and provide publicly available code. In [Kal04], a software tool named ”PHOTOV-III” was developed to determine the required capacity of a PV array and storage for standalone and grid-tied PV systems. This software generates sizing curves but does not calculate the size of PV systems. Moreover, the software does not provide any power flow analysis for the grid connected system. In [EH98], a software program was developed in FORTRAN for calculating the renewable energy source size area based on a well-defined weather profile. The limitation of the program is that it is not user friendly. In addition, no power flow analysis can be conducted for grid-connected systems. Another renewable energy software tool was developed to monitor the performance of a small renewable energy system in a remote area [SB06]. The software is used to monitor PV systems but does not analyze monitored data.

Several commercially available software tools for simulating renewable energy systems and smart grids can be found in [Lal10]. These commercial software tools are RETScreen, PV F-Chart, SolarDesignTool, INSEL, TRNSYS, NREL Solar Advisor

[18] http://www.gridlabd.org

Software	Powerflow Calculation	Source Simulation	PV System Design
RETScreen	-	-	-
PV F-Chart	-	✓	✓
SolarDesignTool	-	✓	✓
INSEL	-	✓	-
TRNSYS	-	✓	-
NREL SAM	-	✓	-
PVSYST 4.33	-	✓	✓
SolarPro	-	-	-
PV DesignPro-G	-	✓	✓
PV*SOL Expert	-	✓	✓
HOMER	-	✓	✓
PV.MY	-	✓	✓

Table 3.3: Comparison of software tools for simulation of PV generation and to design PV systems.

Model, PVSYST 4.33, SolarPro, PV DesignPro-G, PV*SOL Expert, HOMER and PV.MY. These software tools have been developed for analyzing various types of renewable energy systems. Table 3.3 shows a comparison of renewable energy source simulation capabilities for different software tools As indicated, none of the tools is capable of conducting a power load flow for smart power grids. The RETscreen software tool analyzes the economic aspects of renewable energy systems. However, the simulation techniques employed in the software are rather limited. The PV F-Chart software provides analysis for designing standalone renewable energy systems. However, the option of simulating hybrid renewable energy systems in one MG is not included. Moreover, the software provides a monthly performance estimation which may not be as accurate as a daily performance estimation. The SolarDesignTool software provides design recommendations for grid-connected renewable energy systems but it does not predict their performance and impact on the grid. The INSEL software is capable of simulating renewable energy systems but the power flow analysis is not considered. TRNSYS is a simulation program used for simulating renewable energy systems. An example application of the simulation program is performing dynamic simulation of a solar hot water system for a typical meteorological year so that long-term cost savings of such a system could be ascertained. However, the impact of the simulated renewable

energy sources on the power system is not analyzed. The PVSYST 4.33 software is capable of determining the sizing of standalone and grid-connected renewable energy system (but not for hybrid PV systems) and is unable to simulate such systems, whereas PV DesignPro-G only deals with grid connected renewable energy system design. Meanwhile, Solar Pro incorporates calculations that consider shading effects, current-voltage curves and power and financial analysis. PV*SOL Expert provides visualization capability useful for visualizing all roof-parallel and roof-integrated PV systems and also calculates shading on the basis of 3D objects. The HOMER software program is a very good optimization and simulation tool and has been widely used by many researchers. HOMER provides direct optimization for all types of renewable energy systems and performs emission and economic analyses for the designed system. Moreover, HOMER is able to predict the performance of the designed renewable energy system. As with the other tools, the power flow analysis of the MG is not considered. The PV.MY software presented in [Kha12b] calculates the optimal size for renewable energy sources. In addition, PV.MY is able to predict the daily performance of the designed system for one year. However, PV.MY is also unable to conduct power flow calculations for microgrids.

Aside from explicitly dedicated software for renewable energy simulation, there exist power system simulators contributing to this field. GridLAB-D [Cha08], as one example, offers import functions for meteorological data and detailed models for distribution grid devices and residential houses. The current version is, however, completely tailored to the conditions in the US. National and regional differences exist not only in the technical standards and the prevailing grid design. Current and upcoming energy systems (to an ever greater degree) are affected by the local climate, geographical, legal and cultural conditions, as shown in [Mon13]. They determine availability of other (alternative) sources and shape pricing policies and consumer behavior. In the same way, a simulator must be adaptable to the local conditions of the MG being simulated.

3.4 Test Case for Power Flow Analysis

The previous sections provided an overview of the range of functions of different simulation software for power systems and renewable generation. Now, several power system simulators are applied to a reference problem. The results are compared and differences in modeling approaches are described for two purposes: First, to provide a quantitative comparison of the simulators' performance. Second, to identify potential applications in a more comprehensive complex smart grid simulation. Four simulators were selected from table 3.1: MatPower because MatLab is widespread, and is therefore well known and documented; PSAT as it is very complete and provides a lot of data import/export functions; InterPSS promises easyily accessible code, distributed design and good documentation; and GridLAB-D, as the newest simulator with some unique functions, and also highly interesting due to its dedication to distribution system modeling.

3.4.1 The Power Flow Problem

Distribution of power flow and estimation of voltage levels within an energy grid is a popular problem, consisting of several power consuming and/or producing *buses* which are connected by transmission lines, named *branches*. The usual AC power flow analysis considers active and reactive power for each bus as well as a complex voltage values, by magnitude and phase angle. Depending on the type of bus, two of these four parameters are given; the other parameters are estimated by solving a set of nonlinear equations. The equations include the application of Kirchhoff's law and the adjacency matrix from the network of buses and branches. Three different types of bus are common. Load buses with known demand on active and reactive power are the PQ-type. Any power generating bus is mostly of the PV-type and is specified by its active power output and voltage magnitude. Exactly one power generating bus acts as a voltage reference with a given magnitude and a phase angle of zero. This bus is named a *reference* or *swing bus* and its active and reactive power values result from the analysis. The three-phase power transmission lines are usually modeled by a π *equivalent circuit*. The π equivalent model consists of a complex line impedance between the two buses and a capacity to ground. The capacity, specified by its susceptance, is halved to obtain a symmetrically balanced load from both sides of the line. The common three phase system is usually represented with a single line. All branch data are included in the admittance matrix Y, a symmetric matrix with one row for each bus. The power transmitted through, and the losses in the power lines are ascertained by solving the power flow equations. Some branch models further include a transmission ratio and a phase shift to consider voltage transformers.

There exist different simplifications and extensions for the power flow problem. The DC power flow is an approximation which neglects reactive power management and transmission losses [Pur05]. Many applications in energy markets strive to minimize or maximize a cost or welfare function, which is what optimal power flow analysis manages, by considering the physical requirements of the grid. To ensure maximum or minimum thresholds of parameters the bus type can be treated differently. More information about mathematical formulation and solving methods can be found in [Tin67, Mar10] and the book [Mil10].

Power flow analysis is important in order to analyze the impact of the renewable energy sources on MG variables. Analysis is usually conducted using simplified notation, namely a one-line diagram and per-unit system. This analysis aims to calculate the grid bus voltages and phasor angles as well as real and reactive power. Power flow analysis is usually conducted to study the transient states of the MG. Power flow analysis is extremely important for designing a smart grid whereas it is an essential step finding the optimal size and place of any renewable energy-based distributed generation. In the power flow problem, it is assumed that the real power and reactive power are known at each load bus. The voltage angle must be calculated for each generator. In a system with N buses and R generators, there are then

$$2(N-1)-(R-1)$$

unknown variables. In order to solve for these unknowns the power balance equations must be constructed. They are derived by substituting Ohm's law

$$\vec{I} = \mathbf{Y}_{BUS}\vec{V}$$

into the power equation

$$\vec{S} = \vec{V} \cdot \vec{I} = \vec{V}\mathbf{Y}_{BUS}\vec{V} \quad ,$$

both in vector form for the system under investigation. Where $\mathbf{Y}_{BUS}$ is the system's admittance matrix. These equations can be written for real and reactive power for each bus by using:

$$S_i = P_i + jQ_i \quad \text{and} \quad Y_{ik} = G_{ik} + jB_{ik}$$

with some transformation. Here G_{ik} is the real part of the element in the bus admittance matrix, $\mathbf{Y}_{BUS}$, corresponding to the i^{th} row and k^{th} column, B_{ik} is the imaginary part of the same element. Finally, the real power balance equation is obtained,

$$\sum_{k=1}^{N} |V_i||V_k|(G_{ik}\cos\theta_{ik} + B_{ik}\sin\theta_{ik}) - P_i = 0 \tag{3.1}$$

where P_i is the net power injected at bus i and θ_{ik} is the difference in voltage angle between the i^{th} and k^{th} buses while N is the number of buses. While $|V_i|$ is the voltage magnitude at the bus i, θ_{ik} is the phase angle between the buses i and k. As for the reactive power balance equation which can be written as

$$\sum_{k=1}^{N} |V_i||V_k|(G_{ik}\sin\theta_{ik} + B_{ik}\cos\theta_{ik}) - Q_i = 0 \tag{3.2}$$

where Q_i is the net reactive power injected at bus i. These equations describe the real and reactive power balance equations for each load bus. Only the real power balance equation is written for a generator bus, because the net reactive power injected is not assumed to be known and therefore including the reactive power balance equation would result in an additional unknown variable. For similar reasons, there are no equations written for the slack bus. There are several different methods of solving the resulting nonlinear system of equations. The most popular methods are Newton-Raphson method, the Gauss-Seidel method and the fast-decoupled-load-flow method.

3.4.2 MatPower

The main objects in the MatPower model are buses, branches and generators. They provide input parameters to which the power flow commands refer. The bus parameters capture two different types of loads: the usual PQ-type with predefined power demand and a shunt load with fixed admittance to ground. Five bus parameters define the voltage i. e., the base voltage [kV], the magnitude [p.u.], the phase angle and a maximum and minimum threshold for the magnitude. Further bus parameters are a label number, the

bus type, a bus area and loss zone. The bus type is one out of four, an isolated bus is additional to the three common types. The generators in MatPower are allocated to a bus where they feed in the real and reactive power output. Both output values can be limited by a minimum and a maximum value. In addition to these six power parameters (in MW or MVAr res.) is a parameter for the total apparent base load (in MVA). Two parameters for voltage set point and machine status specify emergency exit conditions. Further parameters help to model the interdependences between active and reactive power output more realistically. Possible generator costs are modeled separately. Each branch is specified by the three values of the π equivalent circuit, its pair of buses and a status for in- or out-of-service (to simulate line outage). Optional is a voltage transformation ratio, additional phase angle shift, minimal and maximal values for the angle and three rating parameters.

The default solving method given in the software for AC power flow is the Newton-Raphson method. Further available methods are the Gauss-Seidel and two types of fast-decoupled algorithms. Optimal power flow analysis uses more general solving methods which are available for MatLab. The MatPower manual [Zim11a] gives extensive information about solver methods and methods to extend optimal power flow, e. g., include additional variables. The smart market package and continuous power flow for MatPower described in the technical notes [Zim10] and [Zim11b] were not tested.

3.4.3 PSAT

PSAT uses different user interfaces for model building and analysis. The model setup is done with a Simulink-based editor. Simulink is a MatLab toolbox for modeling of electrical, mechanical or hydraulic systems with a graphical user interface. PSAT provides Simulink libraries for power system elements. The model is built by dragging and dropping components into the model window, wiring them appropriately, and setting each element's properties. The power flow library contains the common elements for power flow analysis, i. e., slack bus, PV-generator, PQ-load, transmission lines (π-model), and shunt loads. Additionally, it includes five different transformers, two alternative transmission line elements, a static condenser (capacity to ground), a static compensator (reactive power generator) and a PQ-generator. The parameter sets are structured similarly for all elements. Their main values are: rating values in SI-units, the nominal values, maximal and/or minimal thresholds, and some checkboxes depending on connection to the grid and changes for exceeded thresholds. A PQ-load, for instance, is specified by rating values for power and voltage, its active and reactive power load, maximal and minimal allowable voltage values and two selectable options: *"Connected"* and *"Allow conversion to impedance for min or max voltage"*. The transmission line parameters are rating values for power, voltage, and frequency; the three values of the π-model; and optionally the length of the line and maximal values for the line's current, active, and apparent power. PSAT uses a special file type to transfer the model data from the Simulink editor.

All analysis is executed from the PSAT main window which provides access to simulation parameter settings, data handling and creation of simple graphs and charts from the results. The default solving method for the power flow problem is the Newton-Raphson methode. Five other methods are available: RX and XR fast decoupled, Runge-Kutta, Iwamoto[Iwa81], and a simple robust method. Further types of analysis are continuation power flow, optimal power flow, eigenvalue analysis, and time domain simulation. PSAT is the subject of several articles, such as [Mil05] and [Van07].

3.4.4 InterPSS

The structure of InterPSS is strictly object-oriented and dedicated to open data model policy according to [Mil09]. Beside power flow, it provides functionalities for analysis of short circuits, transient stabilities, transfer capacities, and power trading. The modeled networks consist of instances from the bus and branch classes. There are five different classes for buses and branches which are related to different types of analysis and inherit properties from each other. For AC load flow, there are specific classes for the network, buses, and branches.

All loads and generators are specified within the bus properties. The possible choices for generator type are the three basic bus types (swing, PV, and PQ), capacitor, non-generation and generator scripting. Capacitors are modeled as reactive power generators. The scripting option allows integration of a custom model. For the load types, there are scripting and non-scripting options, as well as three others to keep either power, current, or impedance of the load constant. Branches can be of type line, transformer, or phase shifting transformer. The branch parameters are three for the π equivalent model, three parameters for rating and optionally a voltage ratio and phase shift for transformers. The selectable power flow solving methods are the Newton-Raphson, Gauss-Seidel, and fast decoupled methods. The selected solver is adjusted by the error tolerance and maximal iteration values.

The distribution system package of InterPSS can perform power flow analysis as well. It is made to model power systems using name plate values from the equipment. Six different types of buses and three types of branches are predefined in this package. InterPSS provides a GUI for data input and network editing. A command line and a grid computing mode are also available.

3.4.5 GridLAB-D

The basic model approach of GridLAB-D is very different from MatPower or PSAT. It models hierarchically structured objects (e. g., at the user device level) which hand over their properties to their parents. A house, for instance, inherits its power consumption from all its dedicated devices. GridLAB-D is module-based, where several modules feature basic simulation functionalities e. g., the runtime module for execution of time

threads and the tape module for recording of time series. The majority of modules define classes, including methods and parameters. The residential module, for instance, defines two different types of houses according to the incorporation of mass effects. There are more than ten parameters defining the shape of a house and even more for characterization of heating and air conditioning. The predefined appliances in the house (e. g., plug, fridge, EV charger) are characterized by three to twelve parameters. GridLAB-D can reproduce complex load scenarios by a high variety of modeled power consumers at a very detailed level.

The network module was made to handle the common AC power flow problem at the transmission grid level. It defines classes for node and link objects and a solver based on the Gauss-Seidel algorithm. A Newton-Raphson solver is part of an update. The node parameters are the parents name, the voltage and apparent power as complex values, node type (=bus type) and a node's name. Optional are an impedance to ground and maximum/minimum thresholds for reactive and/or active power. The main link parameters are the name, a pair of buses (one from bus and on to bus), and the admittance and conductance according the π equivalent circuit. GridLAB-D includes a special power flow module that calculates the power distribution for each phasing line separately, which increases computational and modeling efforts enormously but conforms to the GridLAB-D simulation principle and allows asymmetric and two-phase loads.

3.4.6 Power Flow Results

Power flow analysis results are presented for the IEEE 14-bus test case with four different simulators. This exemplary power system scenario (Midwestern U.S. in 1962) is publicly available[19] in the IEEE common data format for the exchange of solved load flow data. The wiring diagram in figure 3.3 is a screenshot from the PSAT-Simulink editor view. Five of the fourteen buses are at a high (nominal value $69kV$) and nine are at a mid-voltage level (nominal value $13.8kV$, bus nine at $18kV$). This test case has been used in other studies [Kru80, Mil05, Mon85], and input files exist for many simulators, for MatPower[20], for PSAT[21] (including Simulink), and for GridLAB-D[22], with the same files used for simulation purposes. The demonstrated InterPSS results of the IEEE 14-bus test case are sourced from the InterPSS *Loadflow Study Guide*[23].

Figure 3.4 shows the results of the power flow analyses from the four simulators. Figure 3.4a shows the bus voltages (in multiplies of the nominal value) and voltage angles. The PSAT angle values of the middle voltage buses are slightly below the others. PSAT accounts a fixed phase shift of 5 degrees in the branch from bus 4 to 9 which could explain this. The black squares mark all values defined by the input data. The

[19] http://www.ee.washington.edu/research/pstca/

[20] in the package-download *case14.m*

[21] the path *tests/d_014_mdl.m* in the package

[22] in *gridlabd-course-1_1/Solutions/4.1/4_5_IEEE_14-bus_Network.glm*

[23] http://community.interpss.org/Home/user-gude/loadflow-user-guide

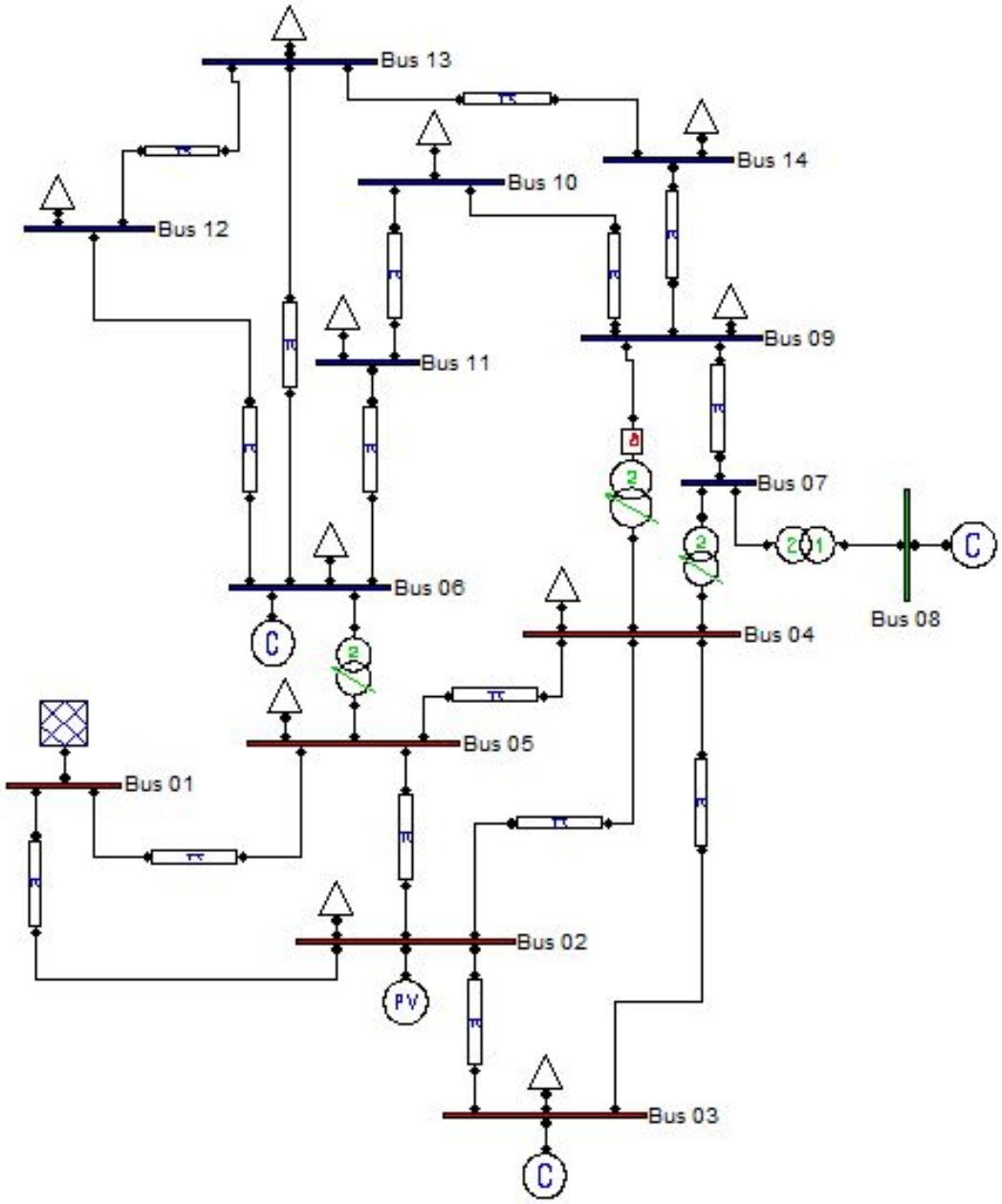

Figure 3.3: Wiring Diagram of the IEEE 14-bus Test Case from the PSAT-Simulink Model. Bus one is the slack bus and is connected to an external power grid. The generator at bus two is the only generator that produces active power, the other three (at buses three, six and eight) feed reactive power into the system. The high voltage level (buses one through five) is separated by three transformer branches from the mid-voltage level. All loads are of PQ-type and the lines are modeled π-like.

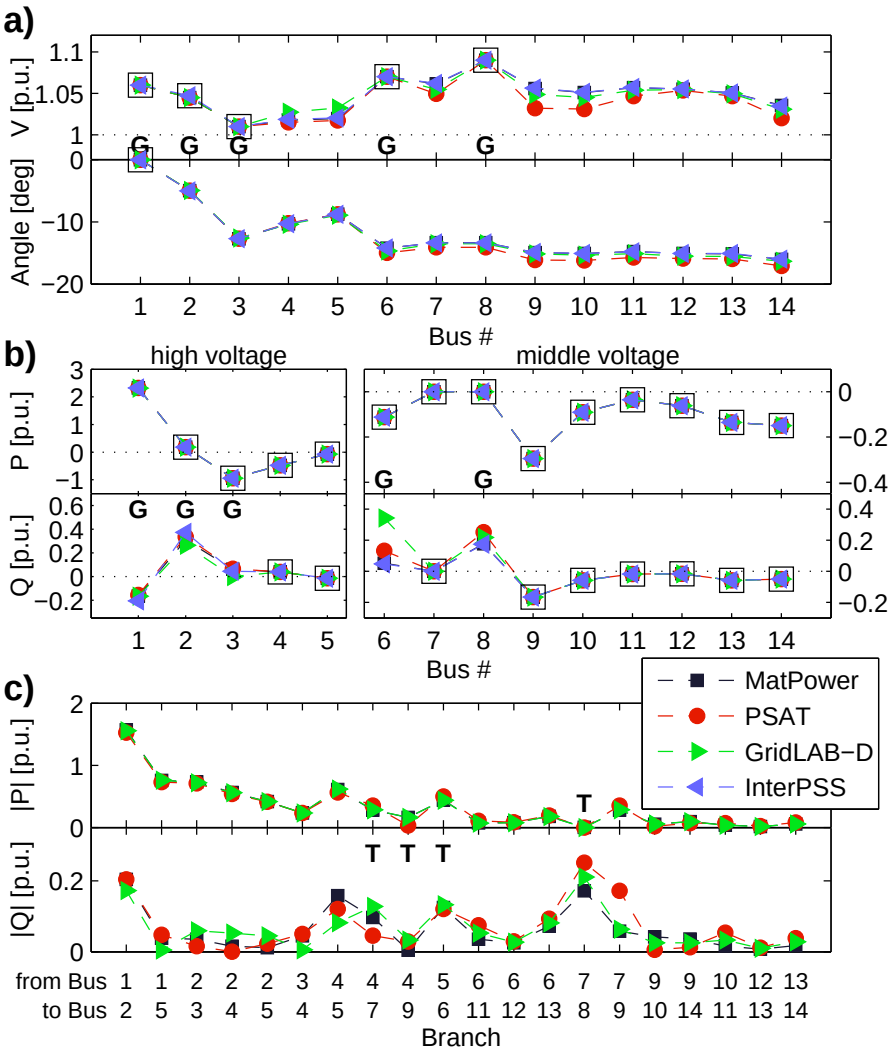

Figure 3.4: Solutions for the power flow problem of four different simulators applied to the IEEE 14-bus test case. The black squares mark the input data for the analysis. Section a) contains the values for bus voltages as multiples of the nominal voltage and voltage angles. Section b) illustrates the real and reactive power at each bus in units of $100MW$ or $Mvar$, respectively. Section c) shows active and reactive power flowing through each branch.

voltage level is known for the generator buses. Figure 3.4b shows the total active power, P, and reactive power, Q, for all buses. The high voltage and middle voltage areas are separated and have different scales. All simulators, except GridLAB-D, follow the convention of showing power producers with positive and consumers with negative signs. The GridLAB-D power results, excluding those of the slack bus, were multiplied by negative one to fit the other data. For this problem, only power for the generator buses and the slack bus power needed to be estimated by the analysis. Figure 3.4c shows the

absolute values of transmitted active and reactive power in each branch for three out of the four simulators. The InterPSS data are missing. The branches which contain a transformer are labeled with a T. The bus of origin is always given a lower number with the destination bus being designated a higher number. The PSAT results were modified to fit that scheme. GridLAB-D provides only the complex line current, not the power. The lines power results from multiplication of the from bus voltage with the conjugate line current as $S_{ij} = -U_i \cdot I_{ij}^*$. The active power results for the branches are quite consistent. The results for reactive power show the most variation between the simulators, which might be related to different the calculation routines for the branch power.

There are additional differences displayed by GridLAB-D. GridLAB-D assumed an additional load at bus nine with a reactance of $B = 0.19$. No transmission ratio specified for transformers and it is the only simulator that uses the Gauss-Seidel method as default. The comparison with the IEEE-14 test case demonstrates that even this relatively simple case can lead to different solutions, depending on the implementation, tiny details in the model, and the solvers used.

3.5 Chapter Conclusion

With the flow network model allows for a more comprehensive overview of the complex topic of smart grids. This principal model is suitable for extension, more detailed itemization, is adoptable to specific cases, and might be helpful for model building. The complex flow network approach considers that the power system of the future will be much closer entangled with the ICT-infrastructure and with energy markets. The established power flow network model is extended by a network of information flow and a network of payment flow. The modeled agents control nodes of the three different networks to optimize their utility function.

Along with this general model, several freely available power grid simulators are compared. Most provided functions are for power systems design. A few simulators provide functions for simulation of energy markets and some for ICT links. The best tools are modularly structured and support partial integration into other software. The latest trends in power system simulation hold some promise but remain far from simulation of the complete complex system of a smart grid which is sufficient to capture interaction with power market issues and changes of user behavior. Another group of software tools, focused on renewable generation simulation, lacks power flow capabilities.

The concluding recommendation is that a simulator of complex smart grid systems should be modular in several senses: It should be openly designed and should use established and tested code as far as possible. It should capture features for market simulation and optimization of welfare and benefit. Any established models should be implementable. Models for user behavior are very rare, indeed the modeling is complicated. There is a strong need for simulating the smart grid in its whole complexity. Not only engineers, but also economists, politicians, and social scientists are interested in

this field. A simulator should be easily operable for all of them, so a GUI, user manual, and proper data compatibility should be mandatory. Recent efforts have built towards this, but a complete smart grid simulation has not been produced yet.

CHAPTER 4

RAPSim - Renewable Alternative Power Systems Simulation

"Art is never finished, only abandoned."

– Leonardo da Vinci

Installation of renewable energy sources in electrical power systems is highly important worldwide due to the fact that renewable energy sources are clean, environmentally friendly, and secure. However, many renewable energy sources such as PV or wind power are dependent on momentary meteorological conditions and are therefore less predictable and/or controllable than traditional energy sources. Consequently, modeling and simulating these sources in the power system is important in order to study their impact on power flow and the quality of the power system [Kha12a].

In order to assess the overall renewable energy production, natural fluctuations can be averaged according to the simulated time resolution. Even though this approximation is sufficient for some cases, it might be necessary to refer to special weather and climate models which are usually not within the scope of power grid simulation. For this reason specific software tools for modeling and simulation of renewable energy sources have been developed separately to address specific problems such as sizing, outcome prediction or economic analysis. A summary of software tools is given in section 3.3. Software tools for (smart) grid simulation, as reviewed in section 3.2, can perform a variety of analyses and functions, but they often lack specific models for renewable sources, especially if the modeled generation unit is small, and no averaging or statistics are applicable, the models must be more precise. There are few precise models for simulating single residential loads. Many common simulation models refer to averaged values from a large ensemble of loads or over long time periods. These statistical models are too large scale to apply to MGs because of the number of entities involved. This is true for loads as well as sources, which means an average household load profile is no longer applicable for a single house with a handfull of residents. These examples demonstrate challenges for the development of future just-in-time and just-in-place energy services which are, according to Ilic [Ili08], part of the energy supply market of the future.

In this chapter, the software tool RAPSim is presented for MG simulation [Las01, Sob12]. RAPSim provides basic models for simulation of various renewable energy sources and load demands within a MG. Moreover, it is able to simulate the performance of the applied renewable energy sources considering some uncertainty of the meteorological conditions. The simulator is further able to conduct power flow analysis for the MG which helps in analyzing the impact of the renewable energy sources on the power system. Finally, the user can easily implement their own models or algorithms, as well as modify existing ones. RAPSim is helpful for simulating smart MGs, and helps to model its components as a part of a self-organizing system [Elm08]. RAPSim can be used to find optimal placement of renewable energy sources in the power system to achieve the best power quality and flow conditions.

A brief list of desirable features for smart MG simulation software is given below. Subsequent sections contain a description of the RAPSim project and its current capabilities. The general description of usage and overview in section 4.2 is followed by a description of the relevant software components for modeling in section 4.3. Specific instructions for implementing a model and examples for renewable production and residential loads are given in section 4.4. In section 4.5, a simple case study is presented, demonstrating the abilities of RAPSim. Finally, some concluding remarks are made, alon with an outlook for future work.

4.1 Requirements for Smart Microgrid Simulation

A list of requirements for MG simulation software is given below, based on the earlier chapters. The list includes different perspectives which refer to the three questions: (A) What functions must be implemented; (B) What can the software be used for; and (C) How can the software be used?

A) Functions

1) **Representation of power grid physics:** In the simplest case, losses are neglected but the power balance equations are satisfied. In further steps, circle analysis and/or the different power flow solving methods should be provided.
2) **Representation of distributed and renewable generation sources:** Renewable sources are crucial components in MGs, for instance, to succeed in carbon gas emission reduction. Therefore a generic modeling structure for renewable sources is required. The general structure helps to design new models or include existing ones. Simulation of renewable sources might require inclusion of local weather and climate data.
3) **Models for residential loads:** Not only sources but also loads must be modeled more precisely. Such models for single households on device level are a necessity for active load concepts, like demand responds.

4) **Capturing grid connected and islanded scenarios:** Some MGs are expected to run in both modes, which requires simulation to cover such eventualities. Additionally, handover scenarios from islanded to grid-coupled mode are of interest.

B) Applications

1) **Teaching and research:** The software tool should be flexible enough to address different research questions and should be easily accessible by novices. Consequently, this requirement has considerable influence on any simulator design.

2) **Playback of measured time series:** A simple way of modeling is the usage of recorded time series of a suitable reference case. Open data policies and smart meter rollout in particular improve data availability. Such data usage enables fast and specific simulation results.

3) **Assessment of device power:** This is a classical engineering question for islanded grids and is essential for related investment decisions. It is what common users are interested in, especially concerning (fuel) generators and storage systems.

4) **Development of MG central controllers:** Assessing different control strategies for a scenario is a classical application for simulation. Therefore simulation scenarios must be repeatable and optionally variable when required.

C) Design

1) **Open Software License:** Publication under an open-source software license is obligatory to reduce economic and legal barriers and reach as many potential users as possible. Free usage of the software enhances further development and increases its overall benefit.

2) **Easily useable:** Simple handling is desireable to minimize barriers to entry for interested non-expert users, as well as novice students without sophisticated programming skills. Tutorials and publications should support user access.

3) **Standard models:** The software should be shipped with a set of different models that can be specified according to different conditions and support the creation of new models.

4) **Depiction of Results:** Simulation results should be easily accessible and shareable. A graphical representation within the software tool would be most desirable. If not, the results should fit a common data processing tool.

4.2 Overview and General Description

The proposed software tool for MG simulation combines models for renewable energy sources and power grid simulation methods. The main area of applicability is considered to be MGs, from the household level up to the distribution grid, on low and medium voltage levels. Many power system simulators analyze the state of the grid under specific conditions and do not require any time specification. For temporal simulation of meteorologically dependent sources, the simulation of daytime and weather conditions is absolutely necessary. This type of temporal simulation is also required for evaluation or calibration of models with measured data.

Naturally, the combination of power grid simulation and renewable source simulation requires a two-level structure: one level for single grid-objects, e. g., a renewable source, and another for the grid itself. Between these two levels it is essential to define an interface clearly so that different models can be easily exchanged. Flexibility of usage of different object-models is further helpful for elaborate load simulation, for instance, the case of a smart house that can perform demand response [Ran10, Mei11], or a charging station for electric vehicles. Many existing tools described in the previous section bring well-developed functionality in their fields. It would be, of course, beneficial to reuse some parts of them if possible. In the best case scenario, a software tool would be able to co-simulate with existing tools. Release under an open software license, as for the software tools reviewed in 3.2, would be highly welcomed by other researchers and the interested public.

The architecture provides flexibility for changing parts of any simulation scenario via the GUI; adapting objects and their models to the local conditions is possible, while the grid-wide simulation algorithm remains unaffected. It allows the inclusion of measured data, e. g., for local weather conditions or user load curves. RAPSim works on two modeling and simulation layers. One layer captures the grid-wide calculations. The second is dedicated to single grid objects, including individual models to describe those objects. The capability of users to implement their specific models increases the range of services and user benefits enormously. This requires Java classes to be written that fit into the structure of the software project. Many functions and features need to be created to simplify this process and make it easier for users with less programming experience. An introduction of the software and a case study were presented in [Pöc14] and [Pöc15], which describes the implementation of user-specific models, including specific examples. The content within this chapter is highly related to these publications which have been written by the author.

All of RAPSim's features are accessible via a clearly laid-out GUI, which is a distinguishing feature compared to many other power system simulators such as MatPower [Zim09], UWPFLOW [Can99], IPSYS [Bin04], MatDyn [Col11], and GridLAB-D[1][Cha08]. As many publications demonstrate the field is rapidly progressing [Wid12],

[1]There exist custom GUIs for GridLAB-D such as GridSpice [And13]. By default, GridLAB-D is distributed without a user interface.

[Sch12], [Pöc13], [Ili08], [Oli12], [Wij13]. RAPSim focuses on MG simulation (island and grid connected) with different types of (renewable) generation and residential loads. It is intended for use in teaching and research.

4.2.1 RAPSim - The Smart Microgrid Simulator

RAPSim was developed at the Institute of Networked and Embedded Systems at the Alpen Adria Universität Klagenfurt. "Raps" is also the German term for the plant *brassica napus* or rape. As the plant contains a lot of energy, rapseed oil is very prominent as alternative, green fuel for engines. RAPSim is meant to be an alternative for green energy as well and it should grow successively. Besides application in science and engineering, the ability to be used in the classroom is essential. The software has a user-friendly graphical interface with direct feedback on simulations being run and support for graphical outputs. Users with no programming skills can use the basic algorithms and models as they are, by accessing them through the graphical interface. In the simplest case the program runs as a Java application, which allows the usage of all the already implemented models and algorithms. Grid objects can be specified, scenarios can be stored and output files can be generated as required by the user. Users with advanced programming skills can add their own models or extend and modify existing ones according to their needs. The programming language for implementation is Java. Thus, the simulator runs on various platforms such as Windows, Linux or MacOS. RAPSim is open-source software so the source code is public and able to be modified by the interested public. Users can import the project into their preferred integrated development environment, for example Eclipse[2]. This enables them to extend the simulator with their own models and algorithms which will be automatically included in the selection menus of the GUI. Further it is intended to keep the software modular and open to use in combination with other tools. This could be specific power system simulation or renewable energy source software, import of meteorological data, an optimization tool or a hardware interface with a laboratory [Mar11a].

Figure 4.1 shows the main simulation window of RAPSim. The basic simulation field is a lattice where objects can be placed, moved, edited, and connected. Below the lattice are buttons that provide some of those functions, with others are accessible through the menu. Grid *objects* are all devices that produce or consume electrical power and help in power transport. All the objects together form a *scenario*, which represents the simulated grid. The objects' parameters can be accessed through the property window, which is opend by double right-clicking. The header of the property window contains the object's identifier or name, which includes its position in the lattice. For instance, the power line at position 14/12 in figure 4.1 connects the buses 2 and 3 and is named "PowerLine @(14|12), Paths: 2<>3". The position is also provided as a tool-tip for the cursor. More complex objects that require specific calculations, such as the power production of a PV generator, own a *model*. The desired model for an object is selectable in the property

[2] eclipse.org

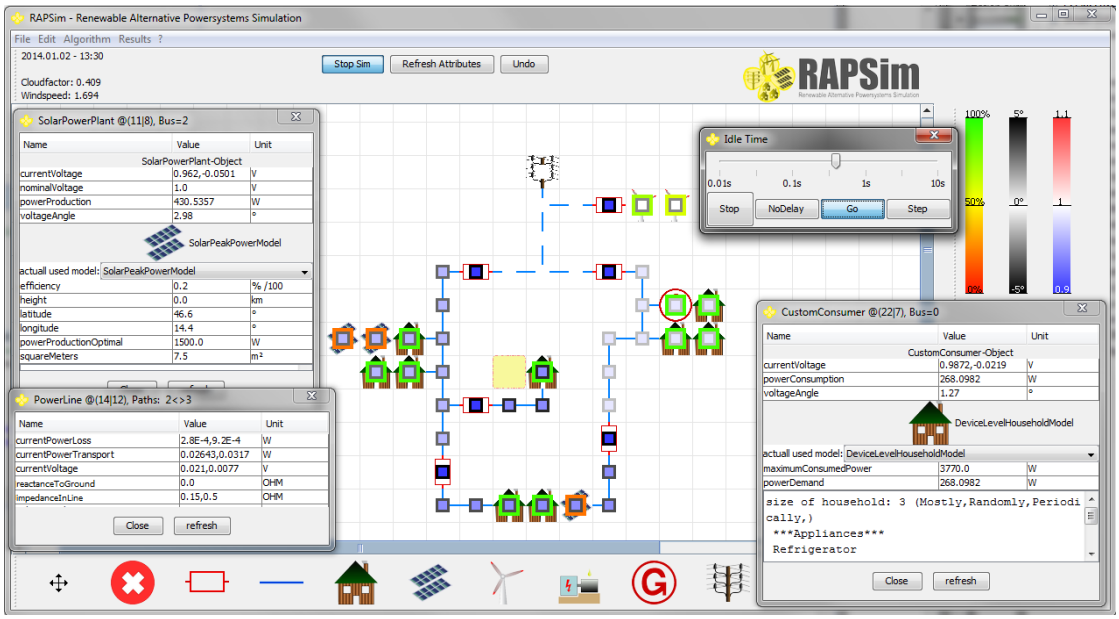

Figure 4.1: The main window of RAPSim shows the lattice-like simulation field where objects can be placed and connected as needed. Object parameters can be displayed and edited in the property window. Color overlays give direct feedback on current simulation values.

window. Further model variables and the model description are displayed, as shown for the solar panels and the consumer in figure 4.1.

RAPSim adds time-based simulation with resolution up to one minute to the static scenario. After pressing the *Start Sim* button the time increment can be specified together with the start date and time and the end point in simulation. The *Idle Time* window is used to define the simulation speed and allows the simulation to be held and step-wise execution to be performed. In figure 4.1 there are three different color overlays activated (the colored squares in the objects). A *color overlay* provides direct visual feedback on the current state of objects by using heat maps. The color overlays show either the fraction of actual power and nominal power value, the voltage angle or the relative voltage level. The running simulation thread provides weather simulation data to feed models of renewable power production. The current weather/time values are shown below the date and time in the top left-hand corner of the main simulation window.

While the model defines the behavior of single objects, grid-wide analysis, and calculations are performed by *algorithms*. Therefore, connected, neighboring objects needs to be grouped. RAPSim does this by generating buses and aggregating parameters from included objects. The bus parameters and paths, which are the connections between the buses, can then be utilized within the algorithms. The algorithm can be chosen via the menu bar. The menu bar provides further functions to edit scenarios, to save and load them in files, and to optionally specify the desired results file. Each object's parameters can be selected by a checkbox to be written into the output file in the CSV format.

The following list contains the features of the simulation framework. The functions a through d are accessible through the graphical interface.

a) A simulation field to create the intended scenarios and to control the simulation functions.

b) Functions to save and load simulation scenarios in a generic XML format.

c) A time thread that models the time of day and day of the year up to one-minute resolution.

d) Generation of output files in the CSV format. All object parameters can be selected to be written into the file at each time step.

e) Weather simulation which can be performed via stochastic models.

f) Topological grid analysis that identifies the objects within a bus and aggregates their parameter values.

g) Administration of algorithms for any grid-wide calculations.

h) Administration of object-specific models that can be easily implemented by the user.

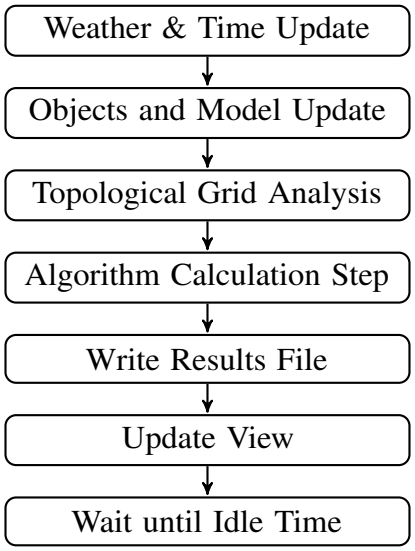

Figure 4.2: These tasks are executed within a simulation step of the time thread. The topological grid analysis is triggered by modifications. The output to the result file must be enabled.

Figure 4.2 shows all the tasks as executed in a simulation step. The weather update in combination with the current time are provided for the following tasks. Many object

models, especially those of renewable generation, are built on these data. In the second step, all the objects are updated. Please note that models must be designed for an adjustable time increment of one minute to provide reliable results. Topological grid analysis is triggered by grid modifications made since the last simulation step. Any grid modification during running simulations interrupts the time thread to redo the topological analysis. User have the freedom to add, remove, modify, or relocate any grid object, even while the time thread is running. The algorithm execution is to some extent the main part of grid simulation. It includes the aggregation of parameters from buses or objects, the execution of the calculation routine and the processing of the outcome. After the algorithm's function has been performed the results are optionally written to the output file, before the view is updated and current values become visible. The purpose of *Idle Time* is to fine-tune the user's view of the simulation. To follow the simulation progress in the GUI it is necessary to pause before the next simulation step is executed.

4.2.2 Simulation Results in File and Color Overlays

Execution of the simulation means the modification of objects' parameters. The related results are therefore part of an object's parameter set and are accessible through the property window. The color overlays give direct visual feedback during running simulations for some special parameters. Results can be saved in a file for storage and further processing. Both functions can be disabled and specified in the GUI, respectively. Color

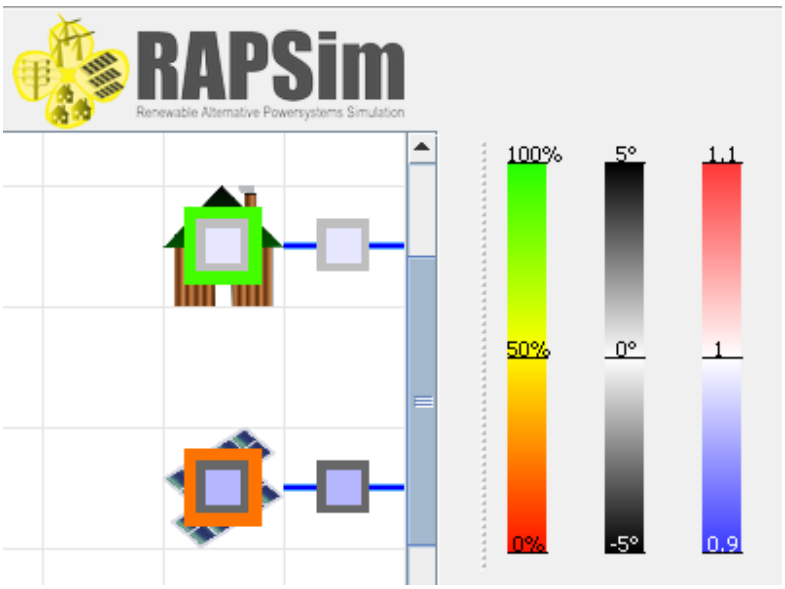

Figure 4.3: Color overlays provide direct visual feedback about simulation results.

overlays should enable users to assess the scenario's current state by showing the values of specific object parameters. Figure 4.3 shows the objects with activated overlays and the corresponding heat-map legend. The overlay values are shown as colored squares of

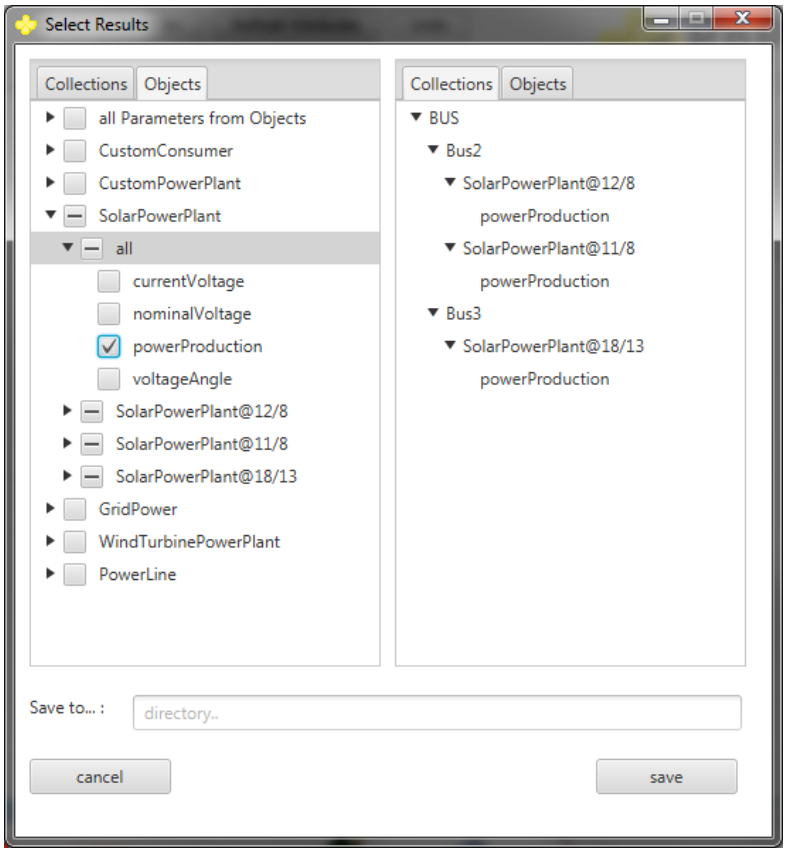

Figure 4.4: The results selection window shows the selection tree, on the left-hand side, and the display tree on the right. All objects' parameters can be selected for the output file.

different sizes on the objects, which means there is a limit (in terms of what is practicable) to the number of displayable overlays. The overlay, with the red-yellow-green image and the large squares, shows the ratio between actual and rated power values. It is implemented differently in the abstract object classes for producers and consumers. The house model, for instance, generates a demand and if fully saturated, as in figure 4.3, the consumption value is the same. For consumers the overlay shows $\frac{P_{Consumption}}{P_{Demand}}$, whereas for generators it shows $\frac{P_{Production}}{P_{ProductionOptimal}}$. The PV panel in figure 4.3 produces around 30% of its rated power. This overlay is not defined for connectors and power lines. The overlay with medium sized squares and the black-white-black symmetric image shows the voltage angle at the object. The overlay with the small squares for relative voltage values uses blue for voltage sags, white for rated values, and red for overvoltage. The limits for heat map scaling can be modified by the user as described in section 4.2.4. The overlays of voltage properties require an algorithm that calculates the related values, as for instance the AC power flow algorithm in figure 4.1. The voltage values are expected to be equal for all objects in a bus, as visible in the connectors of figure 4.3.

To analyze the results of lengthy simulations, it is necessary for them to be stored

in a file. The desired output parameters are selected in the left pane of the results selection window shown in figure 4.4. The right pane shows the list of selected parameters. The selection tree can be enlarged as needed (by clicking the triangles) and enables several fast choice possibilities. For instance, the selection of the power production for all solar power plants in figure 4.4 requires the ”all” checkbox to be checked for solar power plant objects. It is further possible to select ”all” power production variables at once through the item ”all Parameters from Objects”. The display tree shows the selected variables and can be structured independently of the selection tree. Both of them can be sorted by objects (such as on the left in figure 4.4) or by collections, i. e., buses (as on the right-hand side of figure 4.4). The column header in the results file is named as `ParameterName_ObjectType@X|Y-Location_BusNumber` to obtain unique identifiers. For the house in figure 4.1, the header appears as ”powerConsumption_CostumConsumer@(22|7), Bus=0”. Note that the numbering of buses starts at 0 for the first bus. The first column in the CSV file contains the date and time, followed by the cloud factor and wind speed of the weather thread. Column four and onwards contain the values of the selected parameters.

4.2.3 The Weather Simulation

Renewable production, especially wind and solar generation, relies on local weather conditions. If the production devices and related production characteristics are known, a key challenge for simulation is meaningful weather input. One simple way is playback of earlier weather records. This is only possible if records of the required quality are available, i. e., for a given location and for a period and season similar to the simulated one. Another problem can be the granularity or resolution of the data. For instance a PV- production profile cannot reliably be simulated in minute granularity if only daily average values of solar radiation are recorded. Other weather simulation approaches include stochastic random models, which could be then parametrized by measured values. The generation of wind-speed values by sampling a Weibull distribution is one well-known example. The idea of the weather simulation in RAPSim is that several different simulated objects refer to the same weather data, which should reduce the modeling effort for user, while still allowing independent models or specific modification if required.

The RAPSim weather simulation is still in early development, and a data import interface is not yet available. The *weather* class delivers the three parameters cloudiness, wind speed, and temperature. In reality, these are correlated and relate to other parameters, such as humidity or altitude. Due to lack of a complete model for all parameters, they are kept independent. Reproduction of specific results requires the random seed to be equal. It is set to a constant value when the time thread begins.

Temperature: Temperature is directly related to solar radiation (as is PV generation) and is influenced by other factors, such as wind and precipitation. So far, no suitable model for temperature simulation has been found and so it stays at a constant initial value.

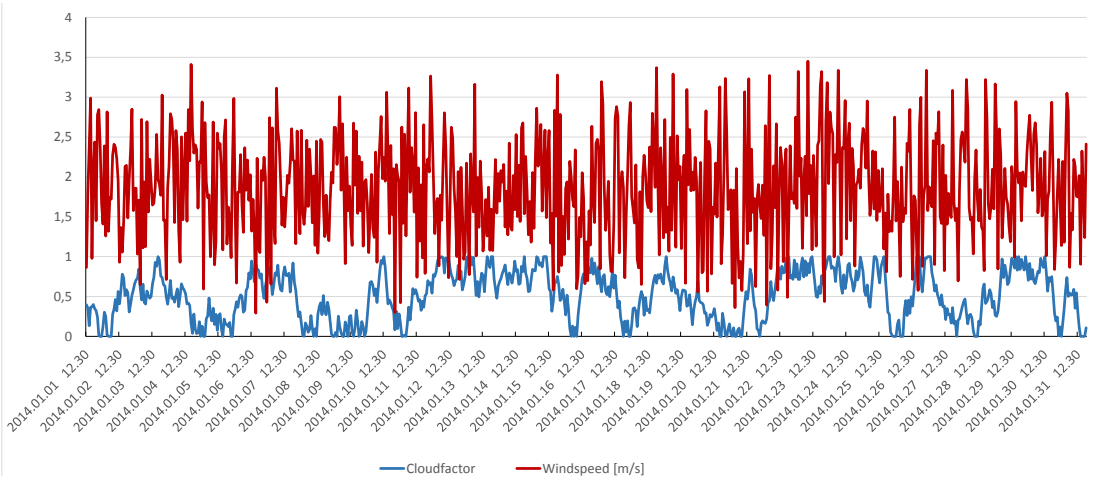

Figure 4.5: Random weather generation samples a Weibull distribution for the wind speed and a random toy model for the cloud factor.

Wind Speed: Wind speed is typically modeled by samples of a Weibull probability density function[Mar11b, Fan11]. This type of random distribution has two parameters which are the shape factor λ (greater than 1) and the scale factor k (between 1 and 3).

Cloudiness: Any cloud coverage decreases solar output and should be taken into account when modeling. The PV-production model uses the clear sky radiation modified by a factor between zero and one for cloud coverage. The design of the simple toy model that generates the random cloud factor was inspired by the measurements of cloud coverage shown in [Boe10, Wau10]. Figure 4.6 shows an example of the cloud factor timelines. The toy model returns a cloud factor $c_t, \in \{0 \dots 1\}$ at every time step t. It includes a scaling factor a and uses a normally distributed random variable ϵ. The shift of the random variable ν starts at 0.5 and is modified according to

$$c_t = c_{t-1} + a(\nu - \epsilon) \begin{cases} \text{if} & (c_t > 1) & c_t = 1, & \nu = \nu - 0.05; \\ \text{elseif} & (c_t < 1) & c_t = 0, & \nu = \nu + 0.05; \\ \text{else} & & & \nu = 0.5 \quad . \end{cases} \tag{4.1}$$

The cloud factor is intended to be used as a random influence for models of PV generation, like the one described in [Pöc14].

The weather simulation parameters can be edited in the class *Weather Model Constants* without reading further code. A desired feature for any simulation of renewable power generation would be the ability to import weather and climate data from public resources on the internet. If such resources are available, the specification of the global position would then be used to obtain realistic weather simulations.

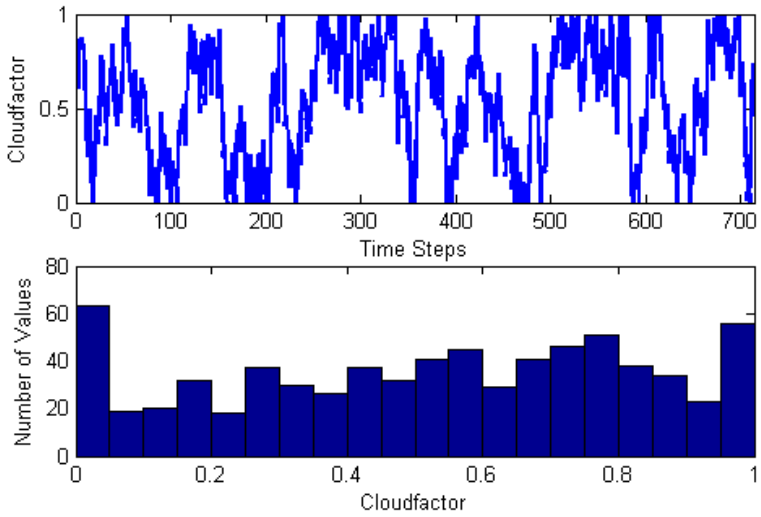

Figure 4.6: The cloud factor is generated by a random toy model. The timeline shows the hourly values for simulation. The histogram characterizes the distribution produced by the implemented random process.

initTemperature: Sets the initial value of the temperature and is set at 20°C as default.

initWindSpeeed: Sets the initial value of the wind speed and is set at 3 m/s as default.

weibullShape: The default value within RAPSim is 3 and is tailored to wind measurements from Klagenfurt.

weibullScale: The default value within RAPSim is 2.1 and is tailored to wind measurements from Klagenfurt.

initCloudiness: Sets the initial value of the cloud factor and is 0.5 as default.

hourlyChange: This attribute represents the scaling factor, a, in equation 4.1 and scales the change of cloudiness per simulation time step.

4.2.4 Customization and General Settings

RAPSim supports the implementation of user-specific models for smart grid objects, which is an important feature for customization. A short overview is given here of further developments which enhance usability of the software within this section. It is very simple to substitute icons for the objects display in the simulation field. All images are referenced by name within the software. Just by replacing one image with another of the same filename in the specific folder, it is possible to attain a different appearance. This applies also to the model icons in the property windows, as well as the edit functions and object buttons below the simulation field. It is highly recommended to use images of the same size.

The class *Program Constants* provides simple access to the parameters concerning program execution. Users can easily modify those values if required. This class holds the following public and static attributes:

dataPath: This attribute specifies the folder where all the icons, representing the smart grid objects in the GUI, are located. For historical reasons the default name of this folder is *Data2*.

defaultSimulationFile: The default simulation scenario, loaded automatically when the software is started, is also located in the folder specified above. The default file name is *data.zip* which can be modified by this attribute.

parameterFile: This attributes specifies a file to store user specific execution parameters, i. e., a Boolean for first program execution and the default name for the results file.

df: The acronym stands for data format. By default it is of type *yyyy.MM.dd - HH:mm* and it is used in the GUI and to write the first column of the results file.

shownDigits: Regularly used double-precision numbers are displayed as rounded values in the property windows. This attribute specifies the numbers of decimal places and is also used for output into the results file.

varCOVoltageAngle: Color overlays allow fast feedback about simulation results by using color maps. The attribute allows to specify the range for the color display of the voltage angle. The entered value, either it is ahead or phase, is shown in black.

varCONominalVoltage: The color overlay for the absolute voltage value is scaled to nominal voltage. White means the nominal value, red means overvoltage and blue, a voltage sag. For instance the number of 0.1 means that voltages between 90% and 110% of nominal value will be displayed.

The class *Program Parameters Saved* defines attributes to manage user-specific execution. There, modification does not require code modification, as attributes are entered during regular program usage or are gathered automatically.

parametersChanged: This parameter triggers the ”save before closing” dialog. It is set to true if changes occurred during runtime that may be saved to the scenario.

firstStart: This Boolean attribute is only set to true the first time RAPSim is run on any computer. It is mainly used to trigger the welcome dialog window.

simResultFile: This string gives the path and file name of the result file. It’s default value is *./simulation.csv*.

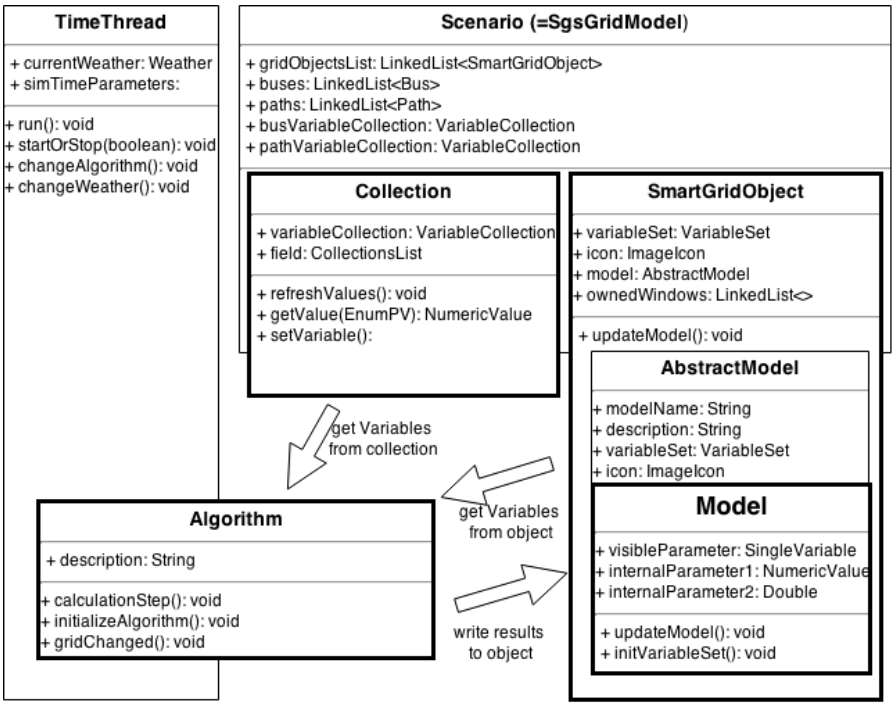

Figure 4.7: Within RAPSim's structure, the Model is encapsulated in the Smart Grid Object. The Objects concern GUI functions and relate to algorithms and collections.

4.3 Software Structure of RAPSim

All classes and packages in the software project are, as far as possible, organized according the model-view-controller (MVC) pattern. The source code is published under an open source license, which allows modifications by the user. Most of the user-specific modifications will likely concern the model classes, as those include the simulated objects. This chapter focuses on that exact part of the software, and the section is more relevant for advanced user than developers. Developers can find more detailed information in the code documentation files generated by Doxygen[3].

Figure 4.7 gives an overview of the main software objects involved in the simulation. The scenario contains a list of all the initialized objects, including their location in the simulation lattice. The network analyzer generates *Collections* for a scenario, which are metaobjects of several grouped smart grid objects. Collections are not visible in the view, nor are they accessible for output. Their function is to support algorithms and simplify their implementation. The scenario is a static group of objects until the simulation is started, which causes the time thread's initialization, which entails the algorithm's creation and the calculation step as shown in figure 4.2.

The following sections aim to explain the concepts and classes needed for implemen-

[3]A tool that generates software reference documentation directly from the code. Available at the website doxygen.org.

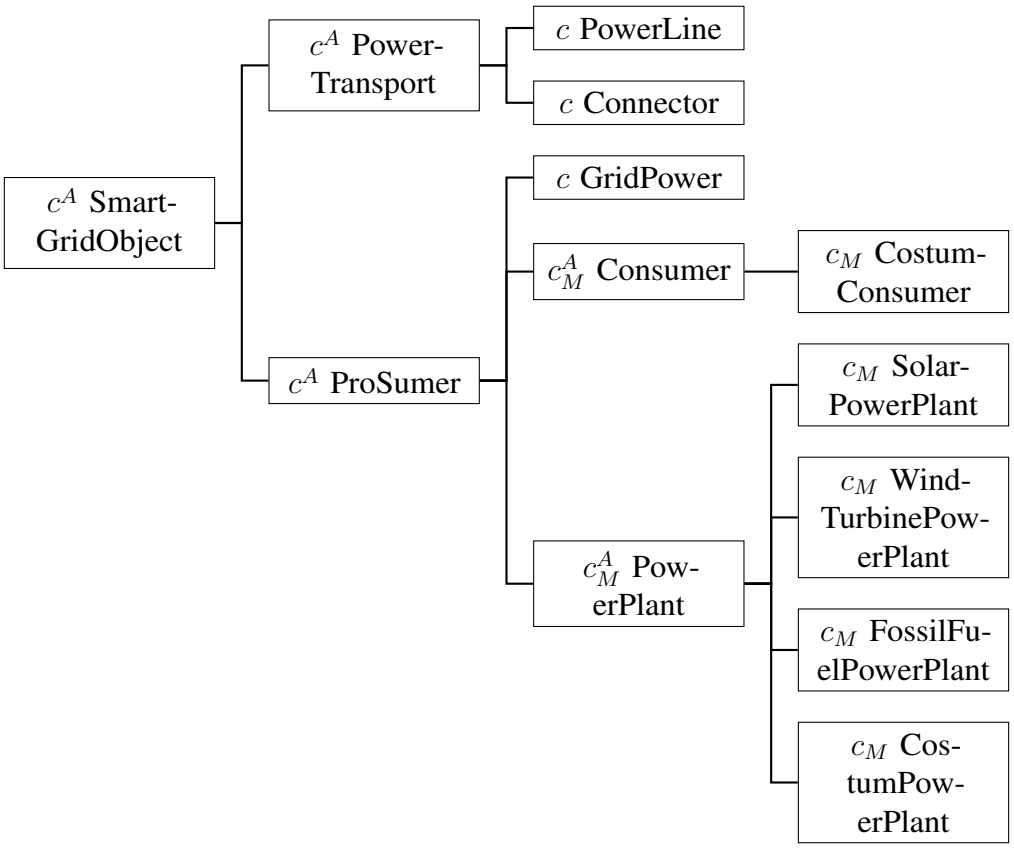

Figure 4.8: The inheritance tree of the grid objects in RAPSim version 0.95.

tation of user-specific object models, including the relation between a smart grid object and the model, the GUI related functions and other recommendations. Another subsection is dedicated to parameter management, which is important for correct depiction and to avoid data type errors. Finally, some general information about and a short description of each of the implemented algorithms is given.

4.3.1 Smart Grid Objects and Associated Models

Any sink or source of power, as well as power transport utilities, is represented as a smart grid object within RAPSim. The smart grid object is the abstract super class for all grid objects. Figure 4.8 shows the inheritance tree for currently implemented smart grid objects. The classes marked with the super string c^A are abstract classes. All objects that require a model are marked as c_M. The abstract object classes implement general functions for:

- Administration of the object's *Parameter Set* and its model,
- handling of the property window,

- generation of the object menu in the GUI, and
- displaying the object icon in the simulation lattice.

The current software structure provides several grid object classes which inherit from the abstract class smart grid object as shown in figure 4.8. This structure will be enriched by additional grid objects when the software project is progressed further. Grid object classes might be added and some which do not yet hold models will do so in future. The type of object implicitly declares some of the used parameters, e. g., all power plant objects have a power production parameter. The specific calculations are usually implemented in the *model* of the object, a protected attribute. Only in simple cases, such as some passive elements, the model is needless. Thus, a model is encapsulated in its object. This enables the usage of different models for the same object, like for instance a wind turbine which can be modeled by the power curve $P(v)$ of a specific type or by its geometry and the kinetic energy in the wind. The model calculates internal behavior, whereas the object communicates its results at grid level via a standardized interface to the enabled algorithm and collections.

Smart grid objects that handle models are associated with an abstract model class. The implementation of new models is simple but requires some Java programming skills. An abstract Java class needs to be extended and the specific methods implemented. A more detailed explanation follows in section 4.4. A newly added model is automatically recognized and integrated in the graphical interface if it extends the correct abstract model class. Figure 4.9 shows the current implemented model structure. Abstract model classes are deduced from the object's inheritance tree (shown in figure 4.8). For all object classes, c_M, there exists an abstract model class which needs to be extended to associate the implemented model with the specific object type. Abstract models organize the interaction with the object and provide general and GUI-related functions for their sub classes. Object-specific parameters (including their set- and get-methods) are defined at the second level, i. e., abstract consumer and power plant model, respectively. Figure 4.9 shows the interface time series model as within the residential average load curve model. This interface is made to be be implemented for any utilization of external data within a time series models.

4.3.2 A Parameter Set of Single Variables with Numeric Values

This section is on the three different classes to administrate parameters for objects and models within RAPSim. Their basic relation is that each *single variable* contains the shown *numeric value* and must be added to the *parameter set*. Together they provide functions for: (i) Visibility and editability of parameter values in the property window; (ii) Handover of values between models, objects ,and algorithms; (iii) Selection and storage of object parameters for the results file; and (iv) Data provision for color overlays. More detailed explanations follow below.

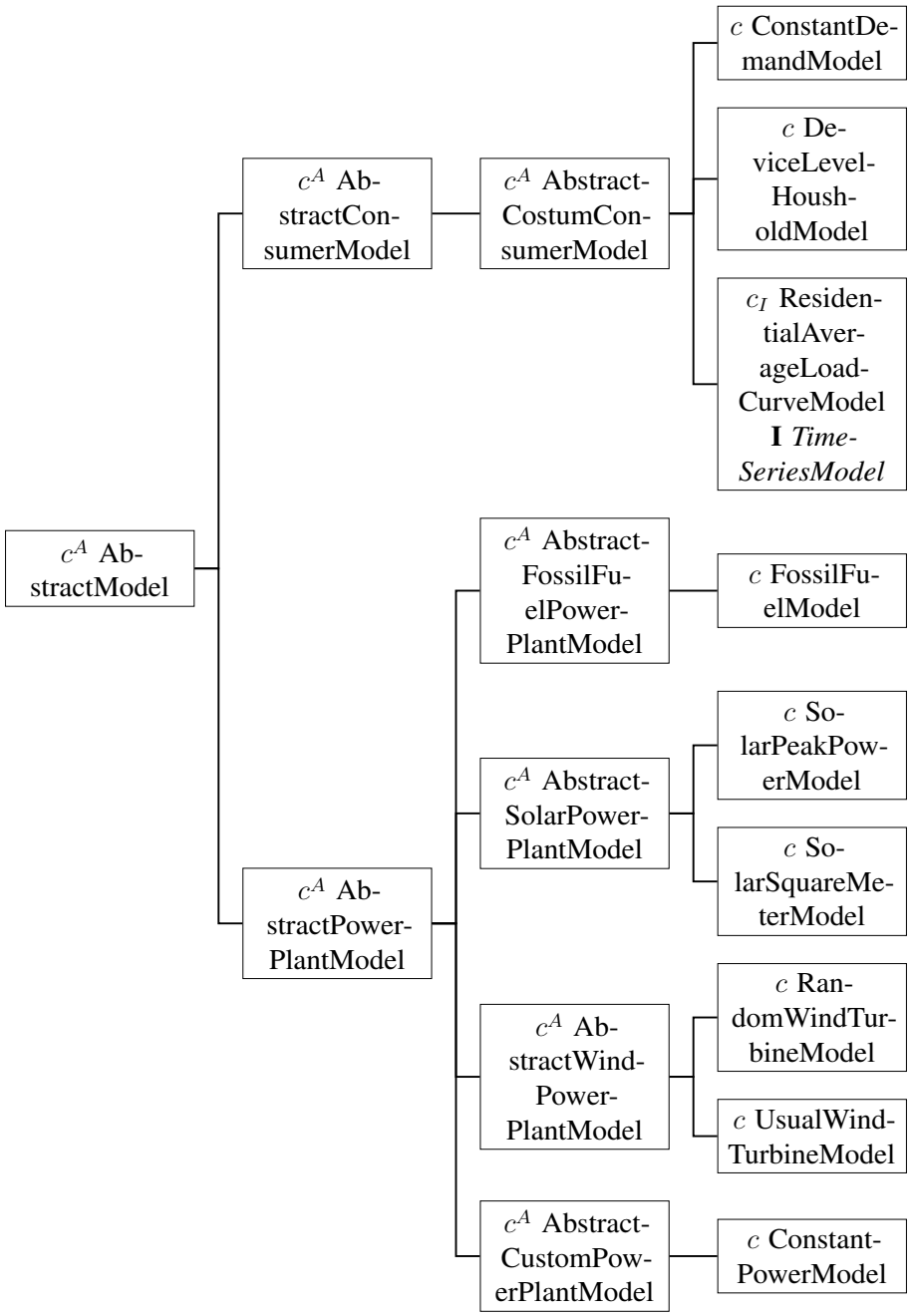

Figure 4.9: Inheritance tree of the model classes of RAPSim version 0.95.

Each smart grid object as well as all the models have an attribute of type *parameter set*. The parameter set manages all the parameters that are visible and optionally editable via the property window of the object. Figure 4.10 shows example of a property window from a wind turbine. The two variables above the icon are part of the parameter set of the object. The visible part of the model's parameter set is shown below the icon. In the current version of RAPSim each object is instantiated with a default variable set as

Name of Variable	Unit	Default Value	Description
powerDemand	Watt	0.0	power demand of component, set by model
powerConsumption	Watt	0.0	power actually consumed, set by algorithm
powerProductionOptimal	Watt	0.0	power optimal produced, set by model and algorithm
powerProduction	Watt	0.0	actual produced power by component, set by model or user
nominalVoltage	Volt	1.0	nominal voltage of prosumer
currentVoltage	Volt	1.0	actual voltage at component
voltageAngle	Degree	0.0	phase angle of voltage, calculated from actual voltage
maxPowerTransport	Watt	0.0	maximum power instance can transport
transportPowerDecrement	None	0.0	percentile decrement for serial resistance
impedanceInLine	Ohm	0.0	complex resistance Z, power line between buses
reactanceToGround	Ohm	0.0	imaginary resistance X_C, power lines impedance to ground
currentCurrent	Ampere	0.0	current over the power line
currentPowerLoss	Watt	0.0	actual power loss on the power line
currentPowerTransport	Watt	0.0	transported power the line

Table 4.1: The parameters values predefined for smart grid objects, separated for prosumers and power transport objects.

contained in table 4.1. Although, dependent on the selected algorithm, many of them are not visible.

To add any parameter to the parameter set, it must be of type *single variable*. It is specified by its name, by its (initial) value, its physical unit, and two options. The two options are Boolean values for setting the variable 1.) visible and 2.) editable in the property window. The name of a model parameter can be any string but should be chosen in a meaningful way and fit the expressions in the model's description. The physical unit needs to be selected from the list *EnumUnit* which can be extended if required. *Numeric*

Value is the obligatory data type for any numbers in single variables. The description for a model is the only exception of a string type parameter as single variable.

Numeric value is a class created within RAPSim to deal with real as well as complex numbers. It provides functions for numeric operations with the same data type and for conversion to the usual numeric Java data types. For common numerical operations, like addition or division, it is necessary to call the specific methods instead of using numerical operators, like + or /. Such operators do not work for the numeric value data type. The class holds a Boolean attribute to mark real values and many methods, for instance to round the value, return conjugate or string-type values, and so on. Many variable values are rounded automatically to be visible in the property window. The responsible method, round value, accesses the *shown digits* attribute, as described in section 4.2.4.

4.3.3 Algorithms and Collections

Algorithms are the main part of the grid simulation. They include all the grid-wide calculations, such as those for power flow analysis. In RAPSim, users can select the implemented algorithms via the main menu, which offers algorithm descriptions also. A newly implemented algorithm is automatically integrated into the GUI if it extends the abstract algorithm class. The algorithm reads parameter values from grid objects and processes them to generate the result, which is then written back to object parameters. Therefore, the algorithm has to enable the respective object's parameters. That means, for instance, that the power flow algorithm sets the voltage variable of a load object. The load object itself does not influence the voltage value at all. The most important methods within an algorithm concern the following functions:

1. Execution of the calculation step as its main function;
2. The setting of properties for the object's parameter values;
3. The setting of properties and initialization of the collections parameter set;
4. Provision of a description.

A *collection* is generally an aggregation of grid objects in a list. The execution of the grid topology analysis generates lists of buses and paths. The path is a single power line object that separates buses from one another. Each bus is a list itself that contains several objects. The collection has a parameter set that holds the aggregated power of all the objects also. There are different functions available for parameter aggregation, such as averaging or summation, used for power values. Collections work, as their name would suggest, by aggregating values from objects, but never write to objects. That means the algorithm can use collections as inputs but must return results to objects directly, as shown in figure 4.7. The buses and paths of the power flow are lists of objects and have a parameter set within RAPSim. The implementation of an algorithm is more

complex than writing a new model for several reasons: Algorithms are more complex by themselves and interact more with other software parts than models. For instance, algorithms set to and get from objects and are closely tied to the time thread execution. The topological grid analysis to identify buses and paths is a valuable service for any algorithm implementation.

The following paragraphs contain a description of the algorithms within RAPSim version 0.95. Each can be used as a blue-print for a custom algorithm.

ActiveReactivePowerbalance: This algorithm calculates a balance equation for active and reactive power of each connected grid in the simulation field. Any losses on the connectors or power lines are not considered at all. The main criterion is the availability of a connection to a superior grid. If one is, the superior connection balances the grid by setting either its production or consumption to the difference between the grid-wide demand and its generation. Otherwise, the consumption or production of grid objects is reduced to balance the total amount of active power and reactive power. When consumption needs to be reduced, consumers are provided with power on a first-come, first-served basis, i. e., later consumers are given no power or a reduced amount of power. The same principle is used for reduction of overall grid generation. Objects are not ordered in terms of their physical location, but by their appearance in the bus grid objects list of the bus (which represents the grid). Reactive power demand is reduced in the same way as the active power. If the scenario lacks explicit reactive generation, the power-producing entities are provided with reactive production according to their share of the total active load.

SimplePowerDistribution: The simple power distribution algorithm could be considered the predecessor of the active reactive power balance. It works in the same way but does not consider any reactive power.

DC Power Flow: DC power flow is a simplified version of the AC power flow algorithm, where ohmic losses on the power lines are neglected. The line impedances cause phase shifts on the buses which are calculated in the algorithm together with the active power. DC power flow is widely used for fast analysis of large systems with up to hundreds of buses. In addition to the results for buses, the power transported by power lines is estimated.

AC Power Flow: This algorithm performs AC power flow analysis (see section 3.4.1) for the simulation scenario grid. It uses a Gauss-Seidel solver. The list of buses and paths is generated by the network analyzer. Each bus is characterized by its i) active power, P, ii) reactive power, Q, iii) voltage magnitude, V and iv) voltage angle, θ, which are contained in the two complex variables (power and voltage). Two of the four parameters are given for each bus, with the other two estimated by the power flow analysis. One bus in the grid is defined as the reference bus or swing bus, i. e., it is the voltage reference with zero voltage angle and a swinging power input.

Therefore, this bus contains the grid connection object. The other buses are either classified as *generator buses*, if the bus is an overall power producer, or as *load buses*. Generator buses imprint the voltage level and their real power production in the analysis, whereas load buses have fixed active and reactive power. For each bus, the two missing parameters are determined by iteratively solving the set of n differential equations $\mathbf{S} = \mathbf{Y} * \mathbf{V}^2$. This equation is in N dimensions, where N is the number of buses, $\mathbf{S}$ is the apparent power $S_i = P_i + iQ_i$, $\mathbf{V}$ is the complex voltage vector and $\mathbf{Y}$ is the admittance matrix. Admittance is the inverse of the resistance and Y_{ij} contains the admittance between bus i and bus j. The admittance matrix requires impedance values for each of the power lines. The power line parameters are, according the π-equivalent circuit, the resistance in line and the impedance to ground. The power frequency must be considered in the input. The solver works with normalized voltage values, which means the reference bus has the value 1. The iterative procedure stops when ether the accuracy of 10^{-6} is reached or more than 100 iterations have been executed.

4.4 Models Implementation in RAPSim

This section completes all necessary information to enable the reader to implement a grid object model in RAPSim. It has been partly published as a conference paper [Pöc15]. The implementation of new models is intentionally as easy as possible and requires some Java programming but does not need sophisticated developer skills. The simplest way is to copy a similar existing model, use it as a blueprint and store it in the same place in the code packages. To fit new classes into the simulator, it is necessary to consider the structure of the software and to follow compatible principles. Therefore, an abstract class needs to be extended and the specific methods implemented. All the classes required for implementation of a new grid object model can be found in the packages `sgs.model.gridObjects` and `sgs.model.objectModels`.

The following section contains a step-wise implementation of a model for wind turbines and descriptions of other models that have potential for modification and extension. The model presented is activated in the wind turbine shown in figure 4.10. It is a mathematical model based on four parameters and an equation which requires the wind speed as an input. The description is explicit from the characteristic equation and follows all the steps until detailed code explanations. The other two presented models are for residential consumers. The first uses time series from an external data source which are adapted to the duration of simulation, as well as the length of the simulation time increment, i. e., the simulation resolution. It constitutes a general example for the import of external data to a RAPSim model. The other model uses statistical data to generate power profiles of appliances in a household. It models on the appliance level and could be suited, for instance, to demand response.

There are three main steps in implementation of a model. The code snippets in

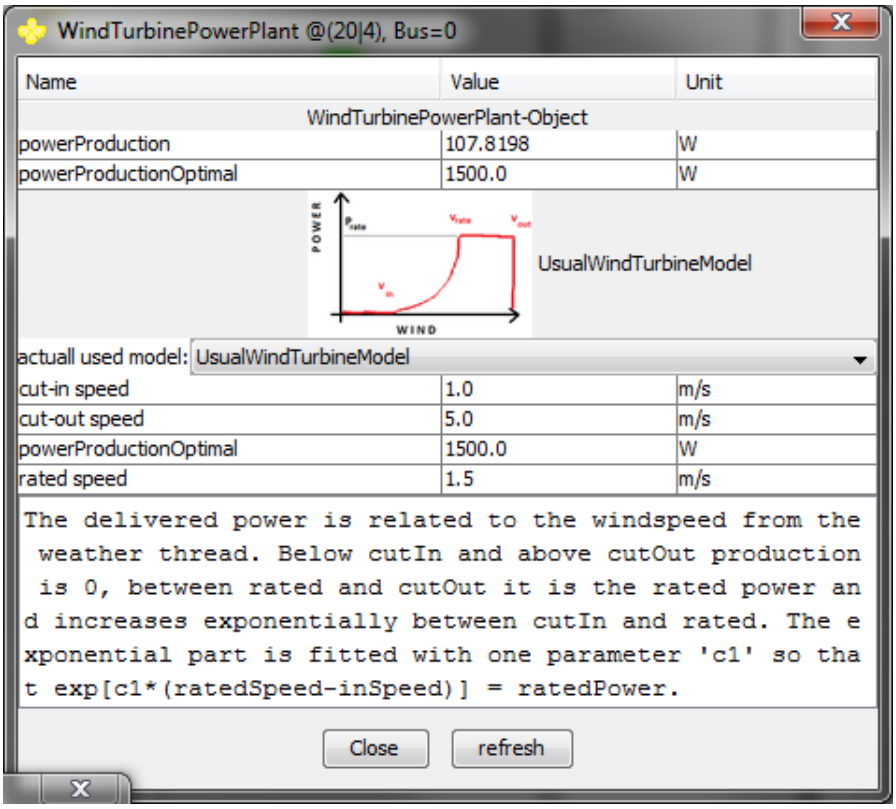

Figure 4.10: The property window of a wind turbine power plant with the *Usual Wind Turbine Model* activated.

section 4.4.1 refer to those steps.

1. Selecting the appropriate abstract model class to be extended and declaring the name, description, and icon for the model (example of code in listing 4.2).
2. Initialization of variables, including the options to be visible and editable in the property window (example of code in listing 4.3).
3. Definition of the update procedure for the variables, which is the core of the model (example of code in listing 4.4).

For each of these three tasks, there is a designated place for the code. The implemented methods are explained with the example of the usual wind turbine model in mind.

4.4.1 Wind Turbine Model

This section describes the model for wind turbines as activated in the property window shown in figure 4.10. The upper part shows two object variables. The algorithm (simple power distribution in this case) sets the required object parameters to visible. The part

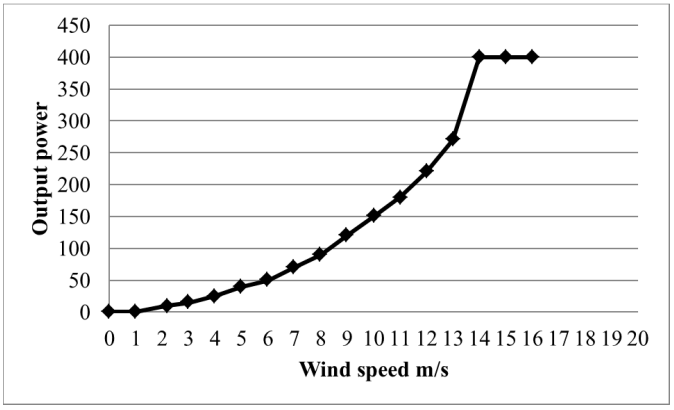

Figure 4.11: An example of a common wind turbine characteristic with 400W of rated power, P_{rated}, and wind speed parameters of v_{cutin} = 1m/s, v_{rated} = 14m/s, and v_{cutout} = 16m/s.

below is designated to the model of the object. It contains a selection menu for the different implemented models, an icon, the variables of the model, and the description field.

The output power of any wind turbine is a function of the wind speed, v. The model uses the values provided by the weather thread within RAPSim. Figure 4.11 shows an example of the power curve of a wind turbine with a rated power of $400\ W$. The power production characteristic, $P(v)$, of this wind turbine model is described by the equation

$$P(v) = \begin{cases} 0, & v < v_{cutin} \text{and} v > v_{cutout} \\ P_{rated}, & v_{rated} < v < v_{cutout} \\ e^{c_1(v-v_{cutin})}, & v_{cutin} < v < v_{rated}. \end{cases} \tag{4.2}$$

where, v_{cutin} is the speed that the turbine starts working at and where v_{cutout} is the wind speed at which the turbine stops working to prevent damage. Between v_{rated} and v_{cutout} the wind turbine generates the rated power. The curve between v_{cutin} and v_{rated} can be generally fitted by any suitable function. In equation 4.2, this is done by one parameter so that $P(v_{rated}) = P_{rated}$ and $P(v_{cutin}) = 1\ W$. The specific parameters depend on the individual wind turbine type. The data can be obtained from the producer or can be measured directly. The implemented fitting function is very simple. A more realistic fitting function with four parameters is, for example, $P(v) = c_1 e^{c_2} + c_3 e^{c_4}$. Obviously, the parameter fitting becomes more difficult and may require a fitting tool based on actual measurements. For simplicity, the single parameter version was chosen for this example. An alternative model for wind generation could be based on a general output formula of the type

$$P = \frac{1}{2} C_P \rho A v^3 \tag{4.3}$$

according to [Cho09, p26]. The equation contains the wind speed v, the swept area of the rotor blades A, the air density, ρ, and the power coefficient of the wind turbine, C_P.

The power coefficient is not usually constant. Consequently, the resulting model is not necessarily simpler than the one in equation 4.2.

The following code listings implement equation 4.2 in Java and perform the functions for model administration in RAPSim according to the list above. The abstract power plant model provides some functions that are needed within the deduced models. For instance, the initialization of the rated power and current power variables. The abstract models also fulfill the forwarding of these two parameters to the object. Listing 4.1 shows the initialization of the variable "rated power" in the abstract model class. First, the parameter is initialized as a single variable with the (initial) value from the power plant object. This procedure allows the model of an object to be changed and to preserve the value for rated power. The other lines of code show the setting of properties and the inclusion into the parameter set (see also section 4.3.2). The default options for variables defined in abstract classes are that visible is set to true and editable is set to false.

```
this.ratedPower = new SingleVariable(
   EnumPV.powerProductionOptimal,
   this.powerPlant.getPeakPower());
this.ratedPower.properties.set(true, false);
this.variableSet.add(ratedPower);
```

Listing 4.1: Definition of the rated power variable in the abstract power plant model class.

Listing 4.2 shows the constructor of the model class usual wind turbine model. Its main content is the definition of the needed attributes, for calculation, as well as for model administration in RAPSim. In addition to the rated power, P_{rated}, the model requires three wind speed parameters, v_{cutin}, v_{rated}, and v_{cutout}. These are of type single variable (see lines 18 through 20), which is a necessity to be visible in the GUI. The fitted parameter c_1 is of type double. The model constructor, line 23, requires the associated power plant object as an argument and calls the super class constructor. Then, the three arguments name, description, and icon are specified. The name should be meaningful for the user but can be any string. The description should not be too long to be visible in the window and should include the same terms used for variable naming. The model icon is visible in the property window and should help to recognize the model. The image must be stored in the folder *Data2* of the project where all other icons are located also. This structure enables simple and fast replacement of images for the simulation. The icon shown in Figure 4.10 is 120 by 80 pixels large and in the PNG format, the required file type and recommended size for model icons.

```
13 public class UsualWindTurbineModel extends
     AbstractWindPowerPlantModel{
14
15 //  is initialized in super class
16 //  private SingleVariable ratedPower;
17 //  private SingleVariable powerProduction;
```

```
18  private SingleVariable cutInSpeed;
19  private SingleVariable ratedSpeed;
20  private SingleVariable cutOutSpeed;
21  private double c1;
22
23 public UsualWindTurbineModel( WindTurbinePowerPlant
      powerPlant) {
24  super(powerPlant);
25  modelName = "UsualWindTurbineModel";
26  description = "The delivered power is related to the
       windspeed from the weather thread. Below cutIn and above
       cutOut production is 0, " +
27    "between rated and cutOut it is the rated power and
         increases exponentially between cutIn and rated. " +
28    "The exponential part is fitted with one parameter 'c1'
         so that exp[c1*(ratedSpeed-inSpeed)] = ratedPower.";
29  icon = new ImageIcon(
       "Data2/WindTurbineUsualModel_ICON.png");
30 }
```

Listing 4.2: The class definition contains the variable declaration and the constructor, which defines the model's attributes and calls the constructor of the super class.

The next code snippet in listing 4.3 shows the method *initVariableSet* which is called in the constructor of the super class. It must still be specified in the model and contains the initialization of the variables and modification of predefined parameter options from the super-classes. First, the options for the predefined parameters are set. The rated power should be editable, although both properties are true on line 34. The power production is set as invisible on line 35 as it is sufficient to see it on the object level. Note that the parameter is still available. On lines 37 through 39, the other model parameters are specified, which are the three characteristic wind speeds defined earlier. This parameter specification is similar to the one shown in listing 4.1, but the three lines of code are aggregated in the single method named *initVariable*. This new method has many attributes, i. e., the name, the initial value, the unit and the two options or properties. The name is the label of the variable as displayed in the property window. It is helpful that the related variable name is easily recognizable and that the format promotes readability. The default value can be chosen freely, including zero, as long as it is of type numeric value. Any initialization of the model is equipped with those values. The unit is required to have a complete variable definition. There is the enumeration list *EnumUnit* where the possible values are stored. The list can easily be extended if required. Both of the options, to set the variable visible and editable, are set to *true* so the values can be modified. If the options are not specified, the default is visible and is not editable.

```
32 @Override
33 protected void initVariableSet() {  // called in constructor
```

```
      of super class
34   this.ratedPower.properties.set(true, true);
35   this.powerProduction.properties.set(false, false);
36
37   this.cutInSpeed = this.initVariable("cut-in speed", new
        NumericValue(1.0), EnumUnit.meterPerSecond, true, true);
38   this.ratedSpeed = this.initVariable("rated speed", new
        NumericValue(3.0), EnumUnit.meterPerSecond, true, true);
39   this.cutOutSpeed = this.initVariable("cut-out speed", new
        NumericValue(10.0), EnumUnit.meterPerSecond, true, true);
40 }
```

Listing 4.3: The method initialize variable set is called by the super class and contains all the variable specifications for the model.

The method *updateVariables*, shown in listing 4.4, contains the models equation. It is executed at each time step within the model update. The method is predefined in the super class including the arguments for date and time, weather conditions and the applied time increment. For this model, only the wind speed is required, which is fetched from the weather thread on line 45. The *if* statement, from lines 47 through 56, implements equation 4.2. Note that usage of the comparison operators requires the arguments to be of the same data type and does not work with the types *single variable* or *numeric values* in RAPSim. The same holds true for other mathematical functions in Java, such as the *Math.exp*. The c_1 constant is recalculated each time step to capture potential parameter modifications via the GUI.

```
42 @Override
43 public void updateVariables(GregorianCalendar currentTime,
      Weather weather, int resolution) {
44
45  double windSpeed = weather.getWindSpeed();
46
47  if (windSpeed<cutInSpeed.getValueDouble() ||
       windSpeed>cutOutSpeed.getValueDouble()){
48   this.setPowerProduction(new NumericValue(0.0));
49  } else if ( windSpeed>ratedSpeed.getValueDouble() &&
       windSpeed<cutOutSpeed.getValueDouble() ){
50   this.setPowerProduction(this.getRatedPower());
51  } else if (windSpeed>cutInSpeed.getValueDouble() &&
       windSpeed<ratedPower.getValueDouble()){
52   c1 = Math.log(ratedPower.getValueDouble()) /
        (ratedSpeed.getValueDouble()
        -cutInSpeed.getValueDouble());
53   NumericValue pValue = new NumericValue(Math.exp
        (c1*(windSpeed- cutInSpeed.getValueDouble()) ) );
54   this.setPowerProduction( pValue );
```

```
55     }
56   }
```

Listing 4.4: The method update variables contains the main calculations for the model.

4.4.2 Residential Average Load Curve Model

Residential loads in the transmission grid are often modeled as averaged load profiles. There are hundreds of households connected to a single distribution feeder. Such models are based on measured data and are used for day ahead planning by grid operators. The German Federal Association of Energy and Water Industries (BDEW) provides average electricity consumption data by a normed German household. The average German load

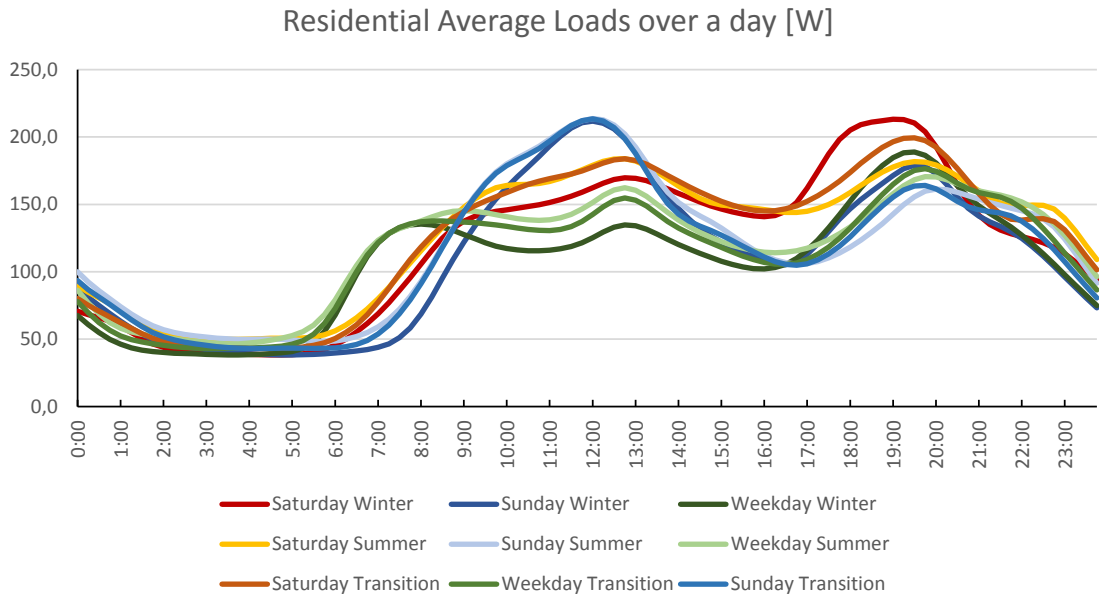

Figure 4.12: The reference Load Profiles for Germany for different seasons and weekdays.

profiles, as used in [Got11], are shown in figure 4.12. The data set includes different profiles for weekdays, Saturdays and Sundays (also valid for public holidays), as well as for the different seasons, i. e., winter, summer and transition (spring and fall). The annual energy consumption of this average profile is scaled to 1000 kWh per year and data values are given in fifteen minute intervals to fit for energy trading.

The data are accessible within RAPSim through the *Residential Average Load Curve Model*. The model plays back the measured data, from a CSV file, scaled by the parameter total *annual consumption*. The data must be processed in the following manner in order to be usable within RAPSim :

- The data file must be readinto an array.
- The correct timeline must be selected for the season and day of the week being simulated.
- The correct values must be selected for the current simulation time.
- The values must be split or aggregated, depending on the size of the time increment used in the simulation.
- The values must be scaled according to the annual energy usage of the house, which is the only editable variable in the model.

The interface, which was used for the presented model, is designed for all time series models in RAPSim. There are many Java libraries which provide functions for the import of CSV files, as time series. In the described model, the Java.io package is used.

4.4.3 Device Level Household Model

Averaged load curves do not give meaningful models for single households. Residential energy consumption is highly fluctuating but still somewhat regular. Many appliances are started directly or operated by humans and their behavior is influenced by regular habits. Some residential demand models use statistical data to simulate reasonable power profiles. In [Got11, Pae13], a load model for households was used to estimate the impact of demand-side management systems. The model must be on the single device level to enable modifications of the single devices' operation time. The *Device Level Household Model* within RAPSim is based on the main ideas of the model used by Gottwalt in [Got11]. Currently, it uses the same statistical data about device penetration in households and residents' distribution. It generates a load curve with step-like variations as produced by a set of devices with different power values. The model is structured as follows:

1. A household is modeled according to its occupants and a set of appliances.
2. The number of occupants is randomly selected and defined, according to table 2 in [Got11], the average annual consumption of the household.
3. There are three different types of occupants who occupy the house for varying amounts of time, according to table 3 in [Got11].
4. Of fifteen types, the appliances are randomly selected according to the parameters in table 1 of [Got11]. The saturation of appliances and the share of the consumption are based on the shown statistical values.
5. Some devices consume power while in standby mode also and some require the attendance of at least one occupant to be started.

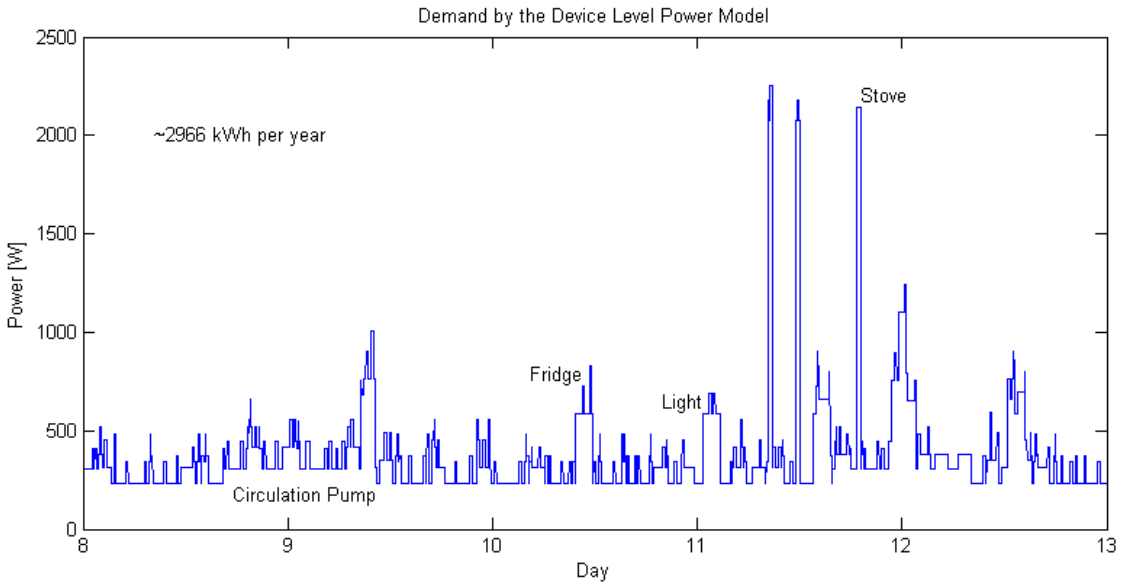

Figure 4.13: Demand of a consumer generated by the device level household model for the period of five days with one minute resolution. Some characteristic switching events are marked with the appliance names.

6. The starting events for each device are randomly sampled. The probability is estimated from the expected annual consumed energy.

7. The standby- power consumption of the devices causes the base load.

Several of the more specific features implemented for the study in [Got11], are not included in the current version of the Device Level Household Model of RAPSim. There are correlations between appliances, occupants' allocations of vacation days, sickness and leisure activities, etc. Substantially, the model runs under arbitrary time increments within RAPSim and is not bounded to fifteen minutes intervals as in [Got11].

The time series in figure 4.13 shows the generated demand for a household made up of one mostly present occupant and a device set that contains a refrigerator, freezer, dishwasher, washing machine, stove, consumer electronics, ICT, circulation pump, light, etc. Some events are marked with the respective appliance name. The standby load is $235\ W$ and the expected energy consumption per year would be around $3\ MWh$. The general data for the devices are stored for input in the file *Appliance.prop*. The statistical parameters for the random construction of occupants are hard coded.

4.5 Test Cases within RAPSim

Within this section are two different test cases presented to demonstrate the expected simulation results with RAPSim. The first is a scenario of time series simulation that

shows the effect of PV production profiles within a grid. The second example is the IEEE 14-bus test case presented in section 3.4.

The scenario with four buses is shown in figure 4.14. It consists of two branches connected to a grid feeder. A photovoltaic generator is introduced which is connected to one or the other of the two branches. The results demonstrate the difference in voltage quality between these two cases of PV integration. The three modeled loads are constant

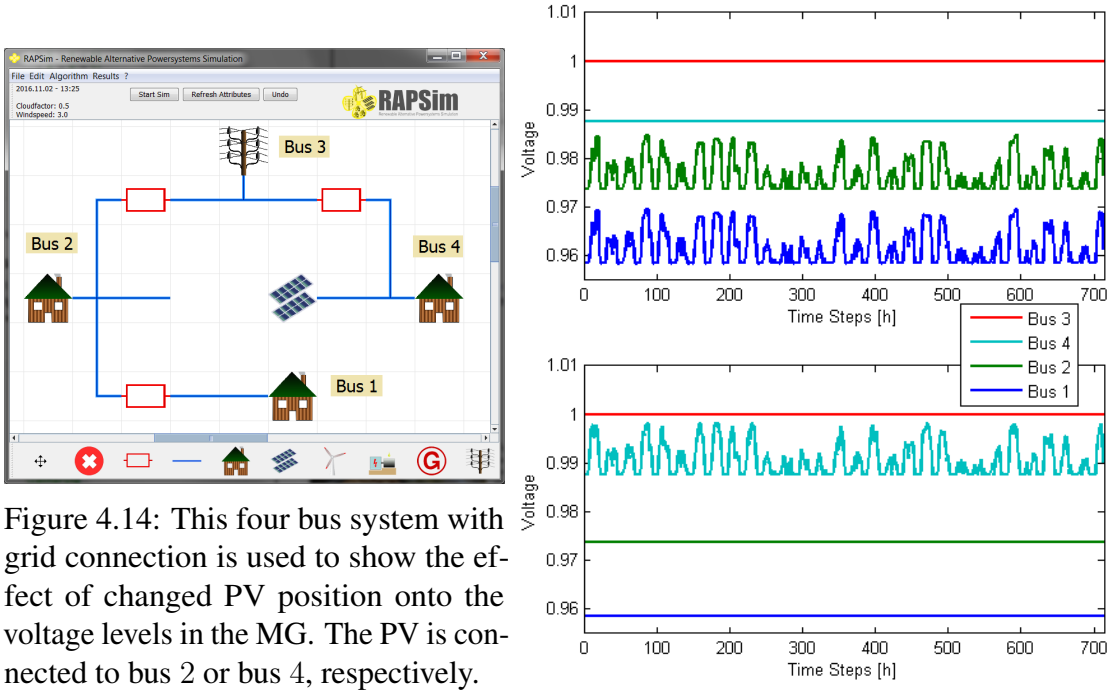

Figure 4.14: This four bus system with grid connection is used to show the effect of changed PV position onto the voltage levels in the MG. The PV is connected to bus 2 or bus 4, respectively.

with a real power demand of 12 kW (Bus 1), 8 kW (Bus 2) and 10 kW (Bus 4). The reactive power demand is set 10% of the real part. The power lines are modeled with a line impedance of $X = 0.1 + \mathrm{i}0.2\ \Omega$ and without capacity to ground. The PV system is specified with a peak power of 10 kW, of active power. The output of the PV system is modulated with a time dependent solar radiation factor while temperature influence is neglected.

The scenario was simulated for 29 days with a 60 minute resolution starting from July 1^{th}. The simulation was executed twice with the different PV connection, i. e., Bus 2 or Bus 4. The random seed for the weather thread was fixed to reproduce the same cloud conditions. The global position of the PV generator is specified by the default values (Latitude 46.6°, Longitude 14.4°) at mean sea level. Figure 4.14 shows the four bus voltages normalized to the slack bus (Bus 3). For the upper graph, the PV is connected to Bus 4, whereas the bottom one shows the scenario as depicted in figure 4.14. The used model does not consider any voltage control unit for the PV system. The produced power is simply fed into the grid and so the alteration in the PV production coincides with the fluctuation of the bus voltages, which are driven by the weather conditions, or more specifically, the cloud factor.

4.5.1 The IEEE 14-Bus Test Case within RAPSim

This test case was used in section 3.4 to compare the results of different software tools for power system simulation. The scenario as modeled within RAPSim is shown in figure 4.15. The presented test case demonstrates the ability of RAPSim to model and

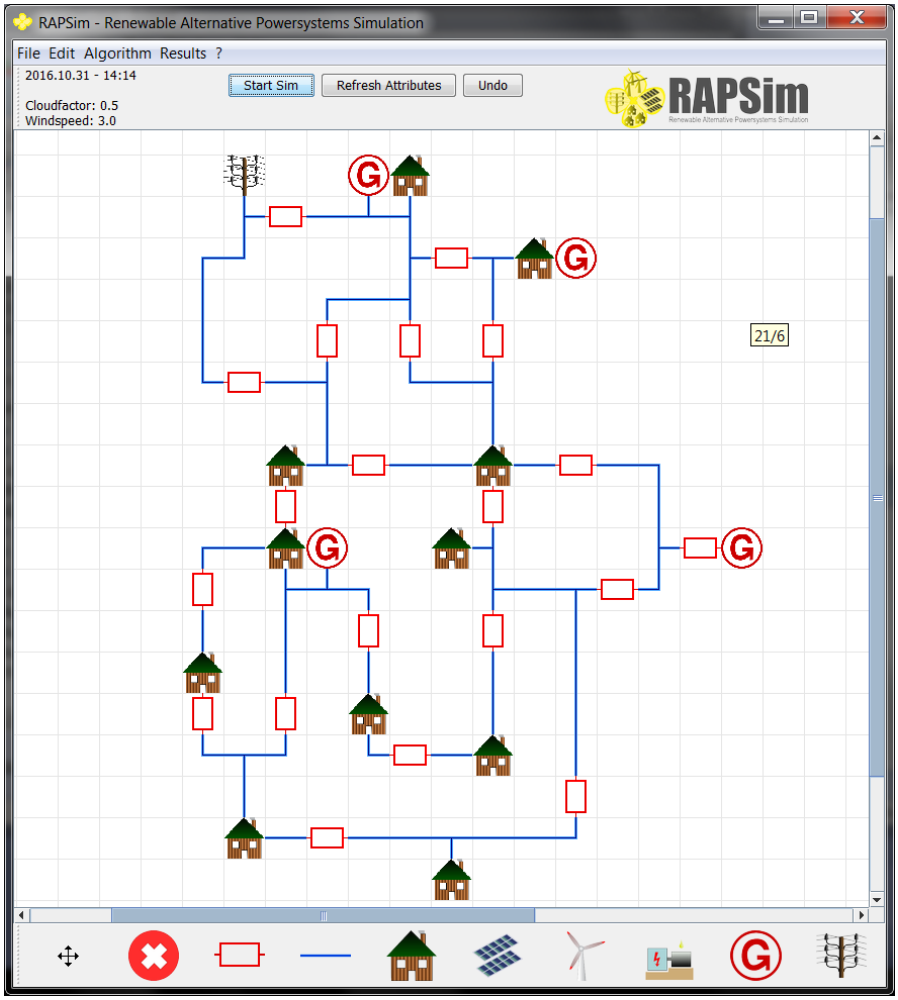

Figure 4.15: The IEEE 14-bus Test Case Modeled within RAPSim.

solve load flow problems. Loads are modeled using RAPSim's constant demand model. The grid connection represents the swing bus and the generators are modeled as constant power producers. As all the objects are modeled with constant power, only a single time step is required to execute the power flow solving algorithm at least once.

The Gauss-Seidel solver in RAPSim requires 298 iterations to reach the required accuracy of less than 10^{-6}. Figure 4.16 shows the results from RAPSim with yellow markers and redraws the content of figure 3.4 (in gray). The results for RAPSim are widely in the range of the other four simulators. Specifically, it must be mentioned that the voltage for bus 7 could not be gathered from the results file, which requires an object

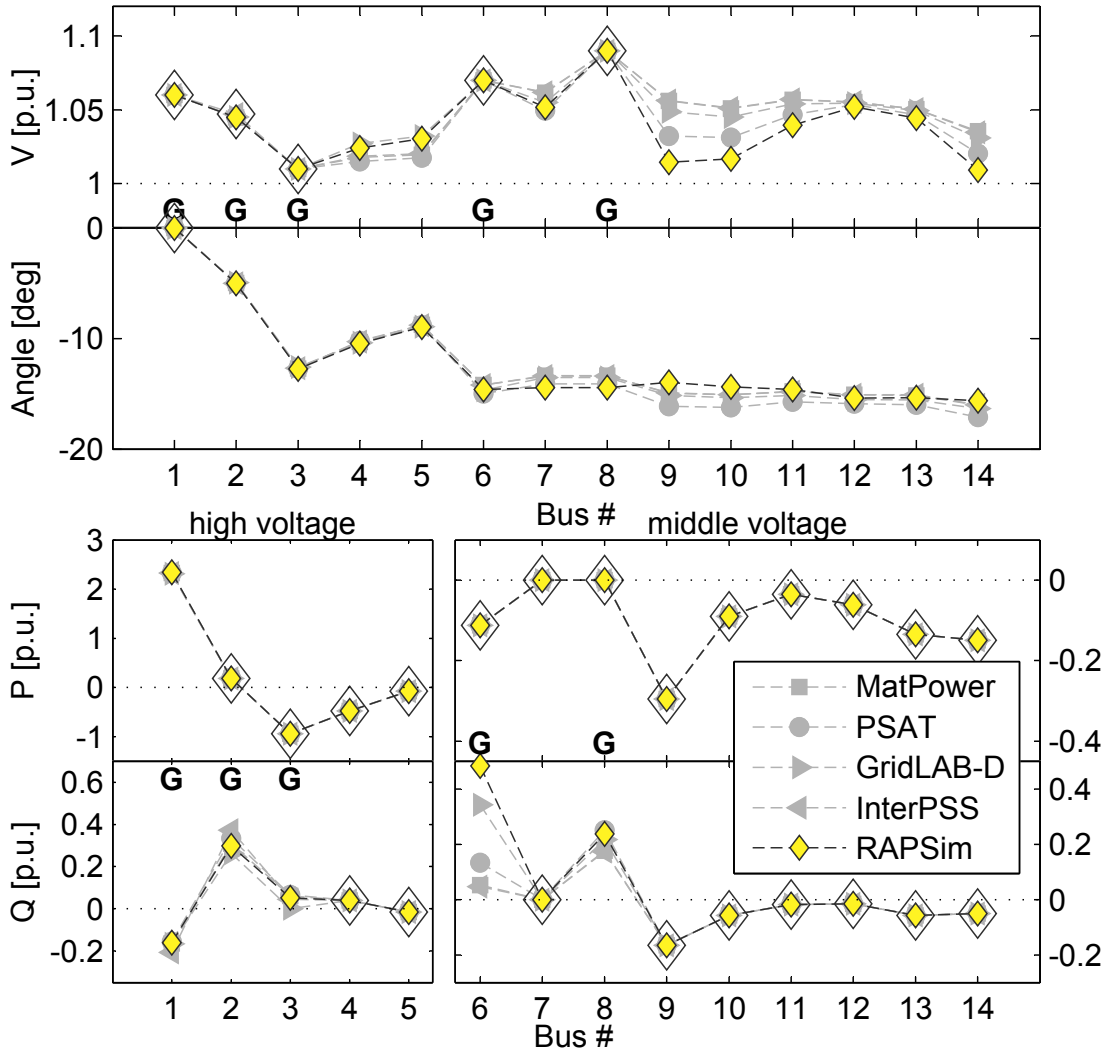

Figure 4.16: Results for the IEEE 14-bus test case by RAPSim compared with other simulators.

that is selectable for output configuration, but connectors are not selectable for output configuration. The possible solution could be either to add a zero consumer or to obtain the voltage from the solver output on the console. Currently, RAPSim has no ability to model transformers or a shunt to ground as it is part of the original test case. For instance, the MatPower test case contains a shunt of 19 Ω at bus 9 which is not considered in the RAPSim scenario at all. The modeling in RAPSim is simple for the user, therefore decisions about power flow solving options are taken automatically. Those assumptions might not fit for all cases which in a sense is a drawback of easy handling.

4.6 Conclusion and Following Work

This section concludes the chapter, with specific reference to the requirements stated in section 4.1.

Functions

RAPSim is designed for time-series simulation down to a one-minute resolution. Its functions include models for renewable generation and residential demand also. The software provides several algorithms that handle different types of grid physics, i. e., power balance, DC-power flow and AC-power flow. The examples in section 4.5 ran smoothly within RAPSim and demonstrated its capabilities in time series simulation, as well as in solving the power flow problem. Different types of power generation are represented by different objects within RAPSim as shown in figure 4.8. This includes PV, wind turbines, diesel generator, and more. The initial software package comprises several models for various objects. These default models may not be a good fit for all use cases. The extension with individual and personal models is supported by the software structure. A detailed description is given in section 4.4. Further models, as described in 4.4.2 and 4.4.3, in combination with the consumer object, represent residential loads in RAPSim. Those models are suitable for extensions and modifications to create new residential load models. RAPSim can be used in simulating grid-tied (as in figure 4.14) and off-grid scenarios. A synchronization scenario as well as AC power flow, in the case of absent grid connection, which represent the swing bus, is not currently in the scope of RAPSim.

Application

An example application of the proposed software is presented in section 4.14, showing the effects of different PV placement on the voltage stability in a MG. The results support the decision about the best placement of the source. RAPSim has already been used for teaching classes on simulation and modeling of distributed systems. Students appreciate the clear layout and simple handling. The research papers [Pöc14] and [Pöc15] based on RAPsim deal with the tool itself. Other research work using RAPSim is known as well, for instance [Übe14]. The average load curve model is a blueprint for the playback of data files within a model. Several GUI-related functions help users to implement their own models and algorithms. The algorithm is the class which a central controller should be implemented. Currently the algorithm executes more functions which could be segmented so that algorithm implementation and controller design is as simple as model creation. It provides functionality for time and weather simulation, for data output and scenario management.

Design

The simulation software has been released as open-source software for all who are interested in smart grids and distributed generation. The public release of RAPSim is hosted at http://rapsim.sourceforge.net and the code can be reused in parts or as a whole. A main emphasis of this system is its comprehensive GUI and easy usability. The graphical interface not only provides editing functions for all objects, but also a

capability for data file specification and selection of different predefined color overlays. The graphical visualization of time series directly within RAPSim is a desired further step. In its current developmental stage, the simulation framework offers a number of basic models. Implementation of further, more elaborate models is continuously ongoing. Currently planned development tasks include increasing the variety of implemented models and the improvement of the weather and climate simulation, including the importing of weather data. The simulation report generated by the software contains power flow analysis of the MG, as well as information about the power generated by the renewable energy sources in the grid.

CHAPTER 5 Information provided by Power Values for Load Disaggregation

"As a general rule, the most successful man in life is the man who has the best information."

– Benjamin Disraeli

The ideas presented within this chapter have been published in [Pöc16], which contains a considerable part of the presented material.

5.1 Introduction to Non-intrusive Load Monitoring

NILM works based on information about the involved devices and permissible assumptions on usage scenarios. This is replicable as power consuming devices unintentionally encode information into the total power draw. Load disaggregation algorithms identify attributes within the measured data and draw meaningful conclusions about the overall consumption scenario. Load disaggregation that works exclusively based on (active) power values is of high interest because active power is simple to measure, and existing metering infrastructure usually provides the necessary values. With smart meters, accessing the data becomes even easier. However, a main drawback is that devices with similar consumption characteristics are hard to distinguish and simultaneously running devices add up in power values. As a consequence the search space of possible power values at least doubles in size with each additional device. Devices with multiple power consumption values further complicate the disaggregation task. A single power value can be either caused by different devices with similar characteristics or by aggregation of multiple less consuming devices. Therefore, the distinction of all possible scenarios using power values exclusively is difficult.

A successfully working NILM-system includes several parts that must be finely tuned. The measurement values must be available with a suitable rate and precision. Knowledge about device set parameters must be managed and provided for disaggregation. Test-cases

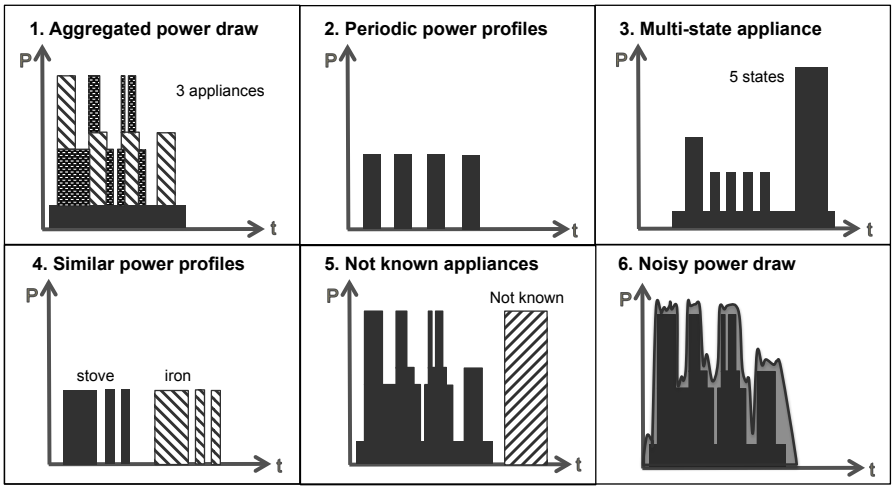

Figure 5.1: Some types of load profiles can cause difficulties for load disaggregation as shown by [Ega15b].

or reference problems for evaluation of the disaggregation performance are required during development. Such an evaluation can become meaningless if the reference problem, often unintentionally, matches the strength of disaggregation approach. Figure 5.1 shows some cases of load profiles that are not straightforward and can cause problems for disaggregation. Whether a test case includes problematic load cases or to what extend it may do so cannot be said easily. Whether a specific case causes problems for disaggregation depends also on the method used.

This chapter discusses the problem of indistinguishable power values caused by different device configurations. Information theory concepts are used to quantize the problem for a given device set by introducing the concept of *proficiency* of power values for load disaggregation. This allows the extent of the problem to be compared more objectively for different device sets. This is done by creating artificial device sets for demonstration purposes and by using real data from different measurement campaigns from houses with multi-state devices. Although only power values are used, the introduced concepts can be extended to capture other device attributes also. The influence of statistical operation probabilities of single devices on proficiency is investigated, and the implications for the improvement of future NILM algorithms are outlined.

The goal of this work is to better understand the mapping of device configuration scenarios to power values. This is identified as a coding procedure for information communication. This knowledge is helpful for further improvement of load disaggregation, which is decoding in that context. The problem addressed in this chapter of the thesis is related to measurement accuracy but it has a very different root and the two problems can be clearly separated. Information theory is used for discrete sources and combinatorics. This work complements others on the limitations of NILM due to measurement accuracy, as well as on quantification of disaggregation complexity for appliance sets.

Section 5.2 is dedicated to explaining NILM as an information communication problem. Within section 5.3 the concepts of information theory are applied to the case of aggregated power values. Two artificial device sets are introduced as examples that contain solely on-off devices. By section 5.4 these concepts are extended to the more general case of devices with multiply power consumption values. In section 5.5, the concepts are applied to nine different appliance sets and they are compared among themselves before the results are discussed and some ideas for future work are suggested in section 5.6. In the last section summarizes the results.

5.2 Load Disaggregation within Information Communication

Load disaggregation was introduced and described in [Har92] by Hart. Figure 5.2 shows a scheme of communication depicting load disaggregation as a decoding problem in the context of information communication theory. Load monitoring benefits from being non-intrusive, which means that any installation or device marking system is avoided. The primary source of information is the appliance usage, which causes power consumption. As the main purpose of a power cable is power supply, the utilized information content is produced unintentionally. The meaningful decoding of a signal stream on the power cable is the challenge of load disaggregation. The code, which is the mapping of the usage scenarios to the power line signals, is exclusively defined by the devices and their attributes. There are various attributes that enable identification of a specific device by its fingerprint on the power cable. Frequency, for instance, and non-harmonic device feedback on the input current is rich on information but the required high-resolution measurement is usually costly and transmission functions of the power line circuits and their influences are barely known. To overcome such difficulties and for other arguments, such as those provided by Hart [Har92] or costs, it is beneficial to use steady-state attributes, such as power values, for device detection.

Several pieces of research have been conducted in relation to the process shown in figure 5.2, such as [Don13], which analyzes limits for scenario detection due to measurement accuracy. Successful and efficient detection of the desired scenario requires the different parts to be well-coordinated. The applications of load disaggregation differ significantly dependent on the acceptable effort on accuracy, maintenance, installation, computing power, measurement, and cost. There are examples of very high measurement rates enabling the distinguishing of quite specific scenarios, e. g., the channel being watched on TV. Thanks to increased computational performance, different device attributes and techniques have been successfully utilized for load disaggregation. However, for many applications, especially for low cost ones, very high data volumes cause a greater burden than they do a benefit.

Current approaches to solve the load disaggregation problem can be divided into supervised and unsupervised approaches. A good overview of supervised approaches

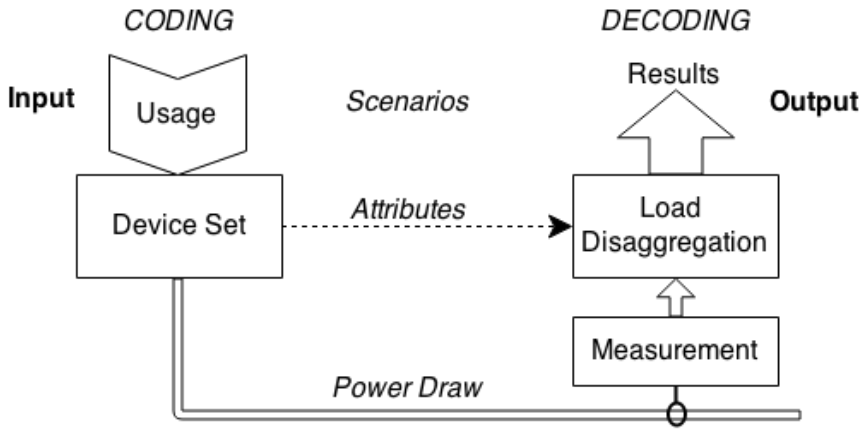

Figure 5.2: Load Disaggregation as the Decoding Procedure in an Information Communication Process.

is given in [Zei11] and [Zoh12]. The supervised approaches require a labeled data set to train a classifier and use either optimization and/or pattern recognition based algorithms [Sha08]. In the optimization-based approaches, the total power consumption and appliance power profiles are given. The composition of the known power profiles with minimal deviation to total power consumption is selected as the current power profile [Lia10, Bar04]. The pattern recognition approaches can be classified into clustering approaches [Har92], neural network algorithms [Sri06, Bie12, Xu15], and algorithms based on support vector machines [Kra12, Alt14]. The disadvantage of the all supervised classification approaches is the necessity of *a priori* information.

Consequently, recent research has been more concerned with unsupervised algorithms, which use unlabeled data. Unsupervised algorithms do not require any training data and therefore no *a priori* information about the system. Current approaches are based on dynamic time warping [Lia14], clustering with blind source separation [Gon11b], Hidden Markov Models (HMM) [Zia11, Pat12, Ega15a], Factorial HMM [Zoh13] other variations of HMM [Kol12, Kim11], temporal motif mining [Sha13], and blind source separation [Gon11a]. For all of these approaches the distinction between appliances is unsupervised, but the labeling of a model with the corresponding appliance is not done automatically. Approaches performing automatic labeling are conducted based on Bayesian inference [Joh13] and semi-supervised classification [Par14].

The device states (specifically their power consumption values) define the set of all possible device configurations, i.e., the state space. The usage is unknown and generates the aggregated power draw. Usage is dependent of the device operators and the built-in programs which make devices change their power consumption. The usage maps the possible device states to power-values. Load disaggregation reverses what usage does: The power profile constitutes input and the current device states can be derived. The mapping of device states to power values is a coding process. The code depends exclusively on the power values of the device states. There is no guarantee that this code

is uniquely decodable and opportunities to modify that code are very limited.

Additional difficulties that arise in the practice of load disaggregation, e. g., measurement resolution or noise, are not considered within this work. The theoretical constraints demonstrated in this thesis arise for an idealized case where integer power values characterize the device states. The explicit inclusion of measurement accuracy is on the one hand not necessary to demonstrate what is aimed for and on the other hand offers no solution to the problem. The basic concepts are elucidated with on-off devices and are subsequently extended to multi-state devices. Correlation between different states and time durations are not taken into account.

5.3 The State Space of an Appliance Set

Within this section, only on-off devices with only one single value of power consumption P_d are considered. The index d denotes the device number, so $d = 1, ..., N$ and N is the total number of devices. The set of devices and so the set of power values

$$\mathbf{P}^D : \{P_1, ..., P_N\}$$

is known. The order of the device set is defined such that $P_d \leq P_{d+1}$. Two or more devices with exactly equal power values cannot be ordered in a unique way and are indistinguishable in the assumed setting. In practice, devices must be distinguished by another attribute. This section uses device set examples with distinct power values, but the formalism is not limited to this case. Examples with equal power values are used in section 5.4 and beyond.

Without any additional knowledge, it is possible to calculate all the possible power values, P_k, by aggregation of consumption values of the device set. The state number k specifies the subset of devices that are turned on and the complementary subset that are turned off. The first state is defined as the power value $P_1 = 0$ and the last state's power value is the sum of all single devices

$$P_M = P_{total} = \sum_{d=1}^{N} P_d \tag{5.1}$$

which can be used to characterize the device set. In between these particular cases are

k	1	2...N+1	...	...	...	M-N...M-1	M
z	0	1	2	...	N-2	N-1	N
n_z	1	N	$\binom{N}{2}$	...	$\binom{N}{N-2}$	N	1

Table 5.1: The table enumerates the M power states, the number of turned on appliances z and n_z, the number of different states with the same z.

always $\binom{N}{z}$ cases where z out of the N devices are turned on. The total number of possible states results in

$$M = \sum_{z=0}^{N} \binom{N}{z} = 2^N \tag{5.2}$$

which is equal to the possible states of a binary word of length N. In the context of load disaggregation, some of those states are very unlikely, or even practically impossible, to occur. Without any *a priori* information about the source and the emitted load profile, it is impossible to detect which states are more likely to occur.

How these M states map to power values depends on the properties of the device set, i.e., the single device power values. The power value, P_k, of a specific state k is calculated by:

$$P_k = \sum_{d=1}^{N} S_{kd} P_d \tag{5.3}$$

where S_{kd} is the state matrix that contains a vector for each state k that holds a 1 for turned-on and a 0 for turned-off devices. Repetition for all the states leads to the set of possible aggregated power values.

Two examples are referred to here of device sets containing ten on-off devices. The device set A has a linear power spectrum, in the sense that

$$P_d = P_{d-1} + P_\Delta \tag{5.4}$$

where $P_1 = P_\Delta = 5\text{W}$. Device set B contains the power values $\mathbf{P}^D : \{1, 2, 3, 5, 8, 14, 24, 41, 69, 117\}$. This power spectrum can be approximated by:

$$P_d \approx \alpha P_{d-1} \tag{5.5}$$

for $\alpha = 1.69$ and $P_1 = 1\text{W}$ and therefore is of the power law type. Additionally these two sets have comparable total power of 275W and 284W, the same magnitude.

A load profile is a stream of power values, P_i, of length n. The total consumed energy is:

$$E = \sum_{i=1}^{n} P_i \Delta t \tag{5.6}$$

where Δt is the sampling time and the power values P_i are averaged over a sampling duration. The average power of a load profile is:

$$\hat{P} = \frac{E}{n\Delta t} \quad . \tag{5.7}$$

The power values, P_i, result from the aggregation of power values of the turned-on devices at time step i so that:

$$P(i) = \sum_{s=1}^{N} P_s S_s(i) \tag{5.8}$$

where N is the number of devices and $S_s^{on}(i)$ is a Boolean state function which is 1 when the device s is operated at the time i and 0 otherwise.

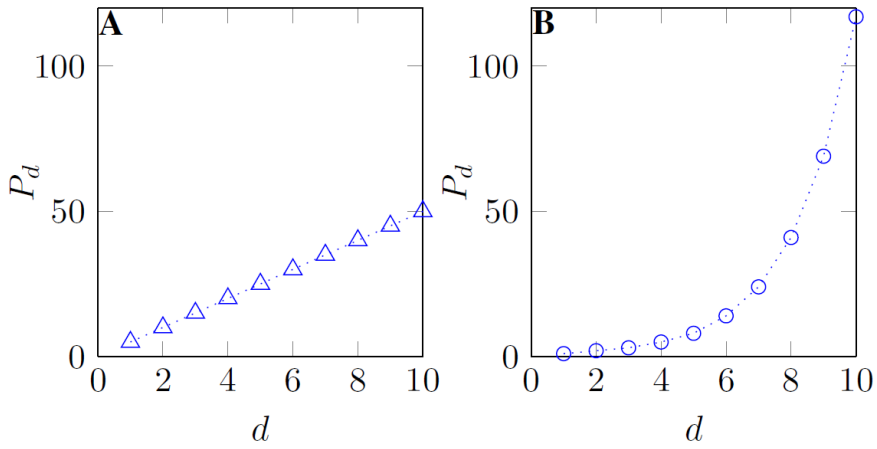

Figure 5.3: Power values of device set A follows equation 5.4 and has $P_{total} = 275$. The set B has $P_{total} = 284$ and single device values according to equation 5.5.

5.3.1 Equal State Probability

When there is no knowledge of a source available it is common in information theory to assume the maximum entropy case, which means an equal likelihood for all possible source symbols. All of the state probabilities, p_k, have the same value of $1/M$ and the entropy of the source, which is defined as

$$H = -\sum_{k=1}^{M} p_k \log_2(p_k) \quad , \tag{5.9}$$

has its highest possible value of $H_{max} = \log_2(M)$. H_{max} is an upper bound for the entropy of a discrete memory-less, time-invariant source and depends only on the number of states, i. e., devices. As a first step, it would allow the comparison of the difficulty of load disaggregation problems with different numbers of devices. Furthermore it is an upper bound for the entropy of any load profile from this source. Table 5.1 shows that there are as many states with one device on as there are with one device off. This leads to the conclusion that if all the M states are hypothetically visited once, each device runs exactly $M/2$ times, half of the total duration. The equal distribution of power state probabilities results in equal average run-time for each single device. This results in the correlation

$$\frac{1}{M}\sum_{k=1}^{M} P_k = \frac{1}{2} P_{total} \tag{5.10}$$

between the average state power and the aggregated power of the device set, P_{total}.

Counting the number of states with a specific power value, P_k, gives a power state occupation number, c. It can be written using the Dirac delta function as:

$$c(P) = \sum_{k=1}^{M} \int_0^{\infty} \delta(P_k - P) dP \quad . \tag{5.11}$$

An occupation number above one reflects the challenge of distinguishing between different states consuming the same power. States with this power value are not uniquely distinguishable. Figure 5.4 shows the occupation numbers for the device set examples which have the same total number of states. For set A there are up to forty states that

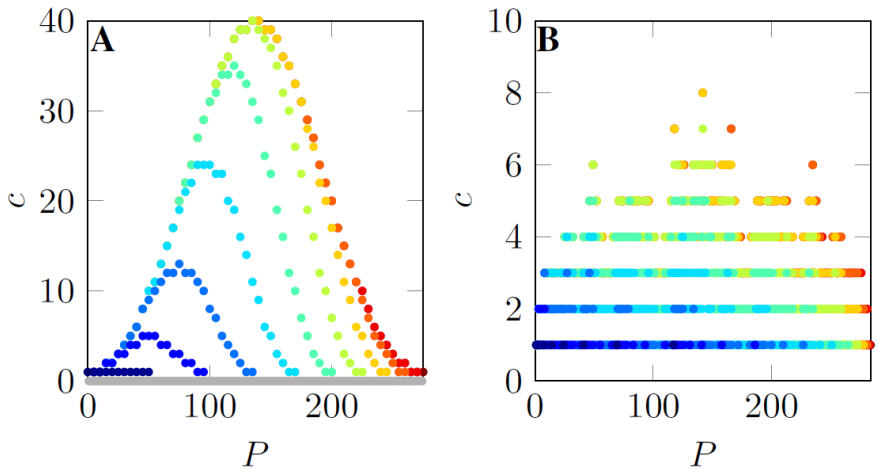

Figure 5.4: Occupation Numbers for Set A and Set B for all Power Values. The colors stand for the number of involved devices starting from $z = 1$ in blue to $z = 10$ in dark red.

map to the same power value, whereas there are only up to eight in set B. In set A, the majority of power values (gray) are not used at all. Load disaggregation is therefore expected to be more difficult for set A than for set B.

The power values in figure 5.4 represent all states of a space between zero and P_{total} which are available to encode the primary information. In that sense, power values are the channel within a theoretical communication setup where the device states are communicated. Otherwise, as for the classical coding problem in communication theory the coding scheme is fixed and cannot be designed according to the channel transmission function. The power values can be seen as the information source of the receiver side. In this context, its entropy is the mutual information of the power value set I^P and is calculated by:

$$I^P = -\sum_{P_j=0}^{P_{total}} p(P_j)\log_2(p(P_j)) \quad . \tag{5.12}$$

The power values P_j are assumed to be a discrete set between 0 and P_{total}, but the definition can be extended to continuous probability density functions. For equal state likelihood, the power value probability is calculated by $c(P_j)$ to allow the following definition:

$$I^P_{max} = -\sum_{P_j=0}^{P_{total}} \frac{c(P_j)}{M}\log_2\left(\frac{c(P_j)}{M}\right) \tag{5.13}$$

which is the information transported by power values for the maximal entropy case. Note that it is not the theoretical maximum of transportable information by these power values.

Therefore it needs the averaged power state occupation number

$$\hat{c} = \langle c(P_j) \rangle \quad . \tag{5.14}$$

On average, each of the $\frac{M}{\hat{c}}$ power states occurs with a probability of $\frac{\hat{c}}{M}$ which can be used further to approximate the mutual information by:

$$I^P_{max} \leq H_{max} - \log_2(\hat{c}) \quad .$$

If the mutual information is less than the entropy of the source, not all information can be transmitted and therefore the stream cannot be decoded completely. The uncertainty coefficient or proficiency is suggested here as a measure of the loss of information. It is defined in information theory [Cov05] as

$$C = \frac{I^P}{H} \tag{5.15}$$

and is shown to be a meaningful performance matrix by [Whi08]. The proficiency for the maximal entropy case C_{max} is given here by:

$$C_{max} = \frac{I^P_{max}}{H_{max}} \leq 1 - \frac{\log_2(\hat{c})}{H_{max}} \tag{5.16}$$

which is restricted by an upper bound that is defined by the average occupation number. Table 5.2 shows the developed information measures mutual information, proficiency and average occupation number for the maximal entropy case for device sets A and B.

	I^P_{max}	C_{max}	$\hat{c}$
Set A	5.33	0.53	18.3
Set B	8.04	0.80	3.6

Table 5.2: The Measures of Average Information for the Device Sets A and B.

For another hypothetical device set, named B2, similar to B with $\alpha = 2$ and $N = 10$, the occupation number is 1 for all power values, as shown in figure 5.7. The set B2 is equal to the binary representation of natural numbers. Consequently, equal probability in state space maps to equal distribution of power values with $c = 1$ which makes the mutual information reach H_{max}. This requires the power values' space to be at least as large as the device state space to enable unique decoding, which means that only in the case where $H_{max} = I^P$ is full load disaggregation by exclusive use of power values possible. The proficiency and the averaged occupation number are both one in this case.

5.3.2 Equal Device Probability

From the point of load disaggregation, it is more appropriate to deal with device probabilities than with probabilities for the combined power states. It is easier to relate devices to different user scenarios than power states. User-dependent devices follow behavioral patterns, e. g., starting the coffee machine after getting up. Automatic devices (such as refrigerators) are turned on regularly and therefore form a major part of the base load in a power draw. For many types of devices, characteristic operation probabilities can be estimated [Kim11]. Even though their occurrence can vary, most are more likely to be switched off. For the sizing of power lines (in a household), utilization factors are standard in engineering. The reasonable assumption is that not all devices (or plugs) are used simultaneously, which allows installation of power lines with smaller cross-sections, which is more economical. Power factors are around 0.5 for households, a little higher for industrial or commercial installations, and they are expected to contain a safety buffer.

However, the state probabilities, p_k, can be easily estimated where the single device operation probabilities, p_d, are known. From equation 5.3, for the state power, the calculation of the state probability, p_k, can be derived as:

$$p_k = \prod_{d=1}^{N} \Big(S_{kd} p_d + (1 - S_{kd})(1 - p_d) \Big) \tag{5.17}$$

assuming that the devices are statistically independent. Specific device probabilities do not fit the maximum entropy assumption as the devices states are not equally likely. For instance, user-driven devices run for a rather short period compared to their downtime. But as there is no *a priori* knowledge of the devices' probabilities p_d, the following expectancy value is used

$$\hat{p} = \langle p_d \rangle \tag{5.18}$$

for all devices, to demonstrate how the device sets entropy is influenced. *A posteriori*, the average probability, $\hat{p}$, for running any device can be calculated by:

$$\hat{p} = \frac{E}{P_{total} n \Delta t} \tag{5.19}$$

using the energy, E, of a load profile of length n. If the single devices' runtimes, n_d, are known, even the device operation probability can be estimated by:

$$p_d = \frac{n_d}{n} \quad .$$

The average device probability is used to obttain:

$$p_k(z) = \hat{p}^{z(k)} (1 - \hat{p})^{N - z(k)} \tag{5.20}$$

which is the state probability of a state with z turned on devices. It is a logarithmic function, as shown on the left-hand side of figure 5.5 for a set of ten devices. The state M,

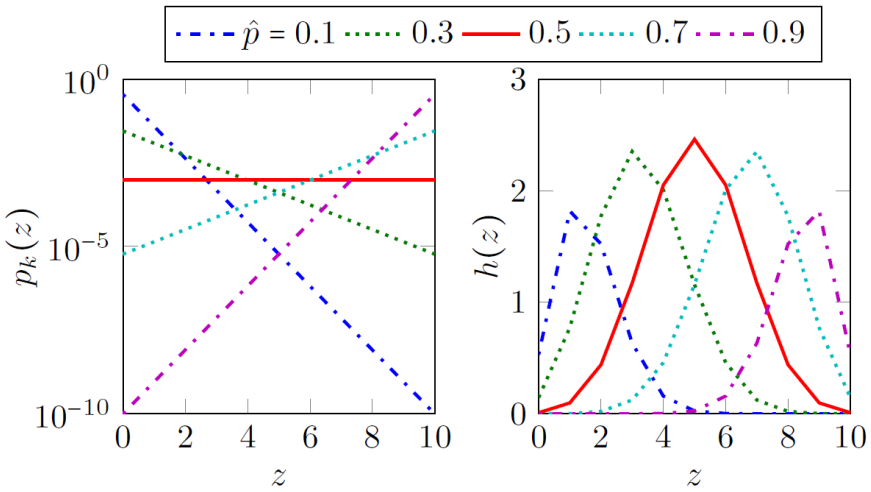

Figure 5.5: The single state probabilities $p_k(z)$ (for z turned on devices) depend on the (averaged) device operation probabilities. The entropy $h(z)$ is additionally determined by the number of states.

with all devices on, has the probability $\hat{p}^N$ and state 1 has $(1-\hat{p})^N$. Figure 5.5 depicts the entropy

$$h(z) = -\binom{N}{z} p_k(z) \log_2 \left(p_k(z)\right)$$

on the right-hand side which is an intermediate result when calculating the total source entropy $H = \sum_{z=1}^{N} h(z)$. In accordance with equation 5.10, the state probability is constant for the device probability of $\hat{p} = 0.5$. The total source entropy, which is shown in table 5.3, then reaches H_{max}. The entropy function $h(z)$ is symmetric with respect to $\hat{p}$, which means the total entropy for an operational probability of 0.1 is the same as a probability of 0.1 of being turned off.

$\hat{p}$	0.1	0.3	0.5	0.7	0.9
H	4.69	8.81	10	8.81	4.69

Table 5.3: Total Source Entropy H for Different Average Device Probabilities $\hat{p}$.

The impact of device probabilities on the entropy propagates to power values, i. e., mutual information and proficiency. The calculation of the power value probabilities

$$p(P) = \sum_{k=1}^{M} p_k \int_0^{\infty} \delta(P_k - P) dP \tag{5.21}$$

requires consideration of the state probability, instead of merely the occupation number,

$c(P)$. This is used to calculate the mutual information

$$I^P = -\sum_{P_j=0}^{P_{total}} p(P_j)\log_2\big(p(P_j)\big) = \sum_{P_j=0}^{P_{total}} h^P(P_j) \tag{5.22}$$

for different single-device probabilities. The function $h^P(P_j)$ is shown in figure 5.6 for the example device sets A and B. It indicates how power values with less involved devices contribute to the overall entropy. The three different values of $\hat{p}$ from figure 5.6

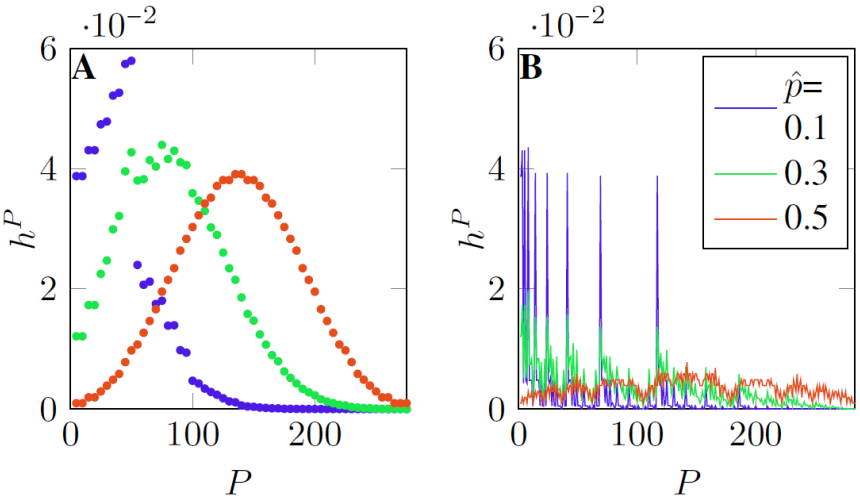

Figure 5.6: The power value probabilities determine the entropy function, h^P, for the power states. Three different averaged device probabilities $\hat{p}$ are shown for sets A and B.

are also used to calculate the mutual information, I^P, and proficiency, C, in table 5.4. Even though the mutual information for $\hat{p} = 0.5$ is higher, the proficiency and therefore the expected accuracy for disaggregation is lower than for $\hat{p} = 0.1$. It is clearly reasonable that equal probability for operation of all single devices does not lead to equal occurrence of the states or power values.

	$\hat{p}$	0.1	0.3	0.5	$\hat{p}$	0.1	0.3	0.5
Set A	I^P	3.70	5.14	5.33	C	0.79	0.58	0.53
Set B		4.50	7.51	8.04		0.96	0.85	0.80

Table 5.4: Mutual Information, I^P, and Proficiency, C, of the device sets A and B, according to figure 5.6.

5.4 Multi-State devices

Multi-state appliances complicate the description of the power values for a set of devices. Power consumption for a single device, d, is specified by a vector with an entry for all its $\hat{s}_d$ power values. A device set with N devices is, for instance, defined by

$$\mathbf{P}^D : \{(P_1^1, P_1^2); (P_2^1, P_2^2, P_2^3); (P_3); \ldots; (P_N^1, \ldots, P_N^{\hat{s}_N})\} \quad .$$

The second device in the example has three power values, which makes $\hat{s}_2 = 3$, and means the device has four possible states. Device three is an on-off device with one power value. The total number of power values

$$S = \sum_{d=1}^{N} \hat{s}_d \tag{5.23}$$

can be used as a characteristic parameter for a device set (in the case of exclusive on-off devices, $S = N$). Just as in section 5.3, all power values are in ascending order to ensure a unique description for a specific device set. The power values of a single device are sorted such that $P_d^s < P_d^{s+1}$. The highest power value of a device, $P_d^{\hat{s}_d}$, defines the order within the device set so that $P_d^{\hat{s}_d} < P_{d+1}^{\hat{s}_d+1}$.

As in the case of on-off devices, the number of possible states is calculated by multiplying of the number of states for all N devices

$$M = \prod_{d=1}^{N} (\hat{s}_d + 1) \quad . \tag{5.24}$$

The M states map to the power values, P_k, which can be calculated by:

$$P_k = \sum_{d=1}^{N} P_d^{S_{kd}} \tag{5.25}$$

where using a different notation than in equation 5.3. The state matrix element, S_{kd}, contains the power state of the device d associated with state k, which is in accordance with its earlier usage. S_{kd} is used as an index, not as an exponent, so $P_d^{S_{kd}}$ is the power value of device d associated with state k. This notation requires the additional definition of P^0 for all devices such that

$$P_d^0 = 0 \quad \forall \quad d = \{1...N\} \quad .$$

The mapping of the state number, k, to the device power state is a little more complicated than for exclusive on-off devices but follows a straightforward principle. For the above example, device 2 is off for the first $\hat{s}_1 + 1 = 3$ states, i. e., $S_{1...3,2} = 0$ or more generally $S_{1...3,d\geq 2} = 0$. For the states $k = 4 \ldots 6$ device 2 runs with its first power value, i. e., $S_{4...6,2} = 1$, and so on. The power value of the last state is

$$P_M = P_{total} = \sum_{d=1}^{N} P_d^{\hat{s}_d} \quad ,$$

the highest possible value. The occupation number, c, is estimated from the set of power values according to (5.11).

The multiple possible device states require a modification of the state probabilities p_k also. The device probability is written in the same way as the power values so that p_d^s is the probability that device d is running on power value s. The state probability is then gained by

$$p_k = \prod_{d=1}^{N} p_d^{S_{kd}} \tag{5.26}$$

using the device state matrix. Additionally the off-state probability, p_d^0, needs to be calculated by:

$$p_d^0 = 1 - \sum_{s=1}^{\hat{s}_d} p_d^s$$

as required with this notation. The notation introduced for multi-state devices is more general and includes the two-state devices from section 5.3. The usage of the average device probability, $\hat{p}$, does not mean an equal likelihood of the devices' power states. It implicitly defines the likelihood of the off-states, meaning the other device states are then equally likely. The state probability is further used to estimate entropy (as in equation 5.9), power value probabilities (as in equation 5.21), and mutual information (see equation 5.22) as in the case of on-off devices.

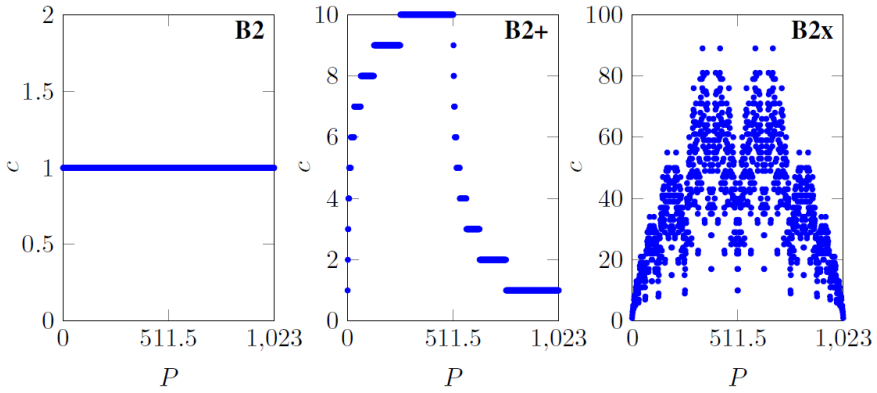

Figure 5.7: Occupation Numbers for Power Values of the Three Artificial Device Sets.

To demonstrate the influence of multi-state devices, three artificial device sets are compared. The multi-state device sets are based on the device set B2, which is constructed the same as set B, but with the parameter $\alpha = 2$, i. e., all states have the occupation number 1, which makes it trivial, as is visible in figure 5.7. The two derivative sets B2+, and B2x both have 9 additional states. For set B2+, device 10 has 9 additional states with the power values of the devices 1 to 9. For set B2x, the devices, excepting the first one, have a second state with the power value of the previous device. The two derivative sets contain the same power values but they are differently distributed among the devices. In B2+, the distribution is very heterogeneous, as only one device has many states. In set

B2x, the device states are almost equally distributed. Figure 5.7 shows the occupation numbers of power values for the three sets. In B2x, the number of states, as well as the highest occupation number, are significantly above the others. The averaged occupation numbers are listed in table 5.5. Table 5.5 contains some more reference values for the artificial device sets of type B. Please note that B2+ and B2x are extreme examples for the distribution of 19 power values among 10 devices. The number of possible states (and the maximum entropy) for any other combination is between the values of those two, independent of the power values. All the type B2 device sets map to the same set of power values and the mutual information does not vary so much. Their distinct proficiency is caused by the growth of the state space.

Device Set	S	M	H_{max}	I^P_{max}	C_{max}	$\hat{c}$
B	10	1024	10	8.04	0.80	3.6
B2	10	1024	10	10	1	1
B2+	19	5632	12.46	9.6	0.77	5.5
B2x	19	39366	15.26	9.8	0.64	38.5

Table 5.5: Characteristic Parameters for the Artificial Device Sets of Ten Devices.

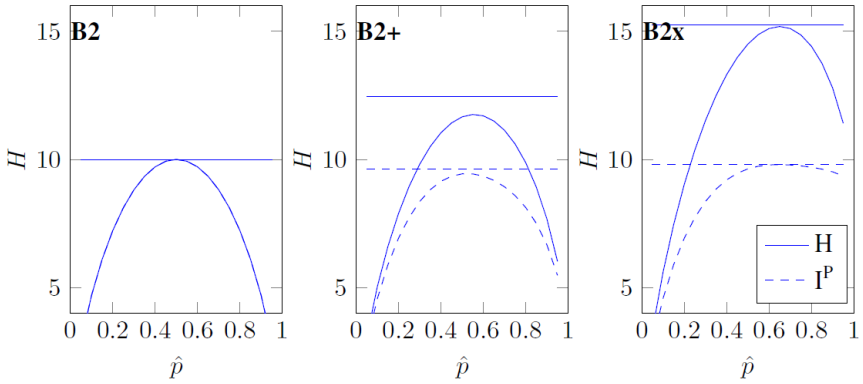

Figure 5.8: Entropy H and mutual Information, I^P, as a function of device probability, $\hat{p}$, for the three artificial device sets. The horizontal lines mark the values for the maximal entropy case.

The dependency of entropy and mutual information on the average device running probabilities is shown in figure 5.8. The horizontal lines mark the values for maximum entropy cases. Dashed lines are used for mutual information. In accordance with table 5.3, the entropy curve for set B2 reaches H_{max} at $\hat{p} = 0.5$. These values shift due to the additional power values in the extended sets. In set B2x, most devices (9 of 10) have two power values, which means they have three states. For nine devices, the equal distribution of states is equivalent to the device probability of $\hat{p} = 2/3$, which is where

the maximum occurs. For set B2+, the entropy function does not reach H_{max}. This is due to differences in the number of power states among the devices. Altough device 10 reaches equal state distribution in $\hat{p} \approx 0.9$, all other two-state devices reach it at 0.5. In other words, device 10 is involved in many of the possible states but is not operated to the same extent as frequently.

The additional states in the derivative sets B2+ and B2x significantly increase the entropy, although the mutual information actually decreases. This is obviously related to the fact that three artificial device sets have the same total power, P_{total}. The decrease in proficiency is caused by extended state space volume of device state combinations.

5.5 Case Study on Real Device Sets

Within this section, the measures for average information, as developed earlier in this chapter, are applied to realistic device sets. Datasets were chosen that have been frequently used as test cases in load disaggregation studies. These are the GreenD [Mon14], RedD [Kol11] and Eco [Bec14] datasets, as used in [Ega15b, Fig13]. To enable device

Device Set	Power States
GreenD 1 ●	[55 140 240], [1220], [60 148 470 570 1225 1265], [1790], [70 155 210 260 423 1898], [40 1900]
GreenD 2 ■	[60], [80], [850], [1580], [80 1725], [90 173 1910]
GreenD 3 ♦	[110 235 285 360], [120 1235], [55 125 540 882 1047 1220 1630], [70 2002], [125 245 358 1998 2100], [70 160 2358 2550]
RedD 1 ●	[200 420], [50 210 410 890 1115], [260 710 1440], [55 110 270 300 620 1405 1505], [1680 2478], [2705]
RedD 2 ■	[123], [410], [160 420], [130 210 770], [1050], [40 1718 1850]
RedD 3 ♦	[100 400], [210 525 730], [40 365 900 1220 1520], [860 960 1285 1605], [120 540 1698], [2265]
Eco 1 ●	[40], [72], [250 440 785], [50 1225], [1800], [90 180 250 365 2168]
Eco 2 ■	[70], [55 175], [80 185], [50 310], [50 1840], [120 2132]
Eco 3 ♦	[100], [120], [130], [100 175 280], [40 1365 1485], [67 190 280 445 650 785 1065 1545]

Table 5.6: Power Values of the Device Sets According to [Ega15b].

set comparability, exactly the same six appliances are used for each house as quoted as submetered power values in [Ega15b]. The power states of the appliance sets, shown in table 5.6, were detected by an algorithm presented there. For further, more detailed

Device Set	$S+N$	M	H_{max}	I^P_{max}	C_{max}	$\hat{c}$
GreenD1 ●	26	2352	11.2	10.21	0.91	1.23
GreenD2 ■	15	192	7.59	7.20	0.95	1.10
GreenD3 ♦	30	10800	13.4	11.69	0.87	1.76
RedD1 ●	26	3456	11.75	10.72	0.91	1.18
RedD2 ■	17	384	8.59	8.4	0.98	1.94
RedD3 ♦	24	2880	11.49	10.04	0.87	1.67
Eco1 ●	19	576	9.17	8.84	0.96	2.24
Eco2 ■	17	486	8.92	7.86	0.88	1.79
Eco3 ♦	23	1152	10.17	8.97	0.88	2.57

Table 5.7: Parameters for the Nine Sets of $N = 6$ Devices. Comparison is shown in figures 5.9 and 5.10.

information, e. g., a method for extracting appliance state information and the choice of appliances, please consult [Ega15b].

All parameters shown in table 5.7 result directly from the power value sets in table 5.6. The values are presented in figure 5.9, which shows the houses ranked according to their number of states, and in figure 5.10, in which they are ranked by descending proficiency. Figure 5.9 depicts entropy and mutual information for the maximal entropy case and characteristic power values, i. e., the total set power P_{total} and average device set power, P_{av}, of the sets (kilowatts). P_{av} is the expected average value when all devices are turned on. It is calculated by obtaining the average power for each device

$$\langle P_d \rangle = \frac{1}{\hat{s}_d} \sum_{s=1}^{\hat{s}_d} P_d^s$$

and calculating the expected value for the device set

$$P_{av} = \frac{1}{N} \sum_{d=1}^{N} \langle P_d \rangle \quad .$$

Statistically, datasets with more states are expected to have higher total power values. GreenD2 and Eco3 are exceptions, which leads to the conclusion that bias in device selection can be significant. Conclusions about the number of states using the total power is inappropriate for individual device sets. The average set power is between 50 and 75 % of the total set power. A general statement about the characteristic power values of real device set would need a separate study with many device sets and different types of appliances.

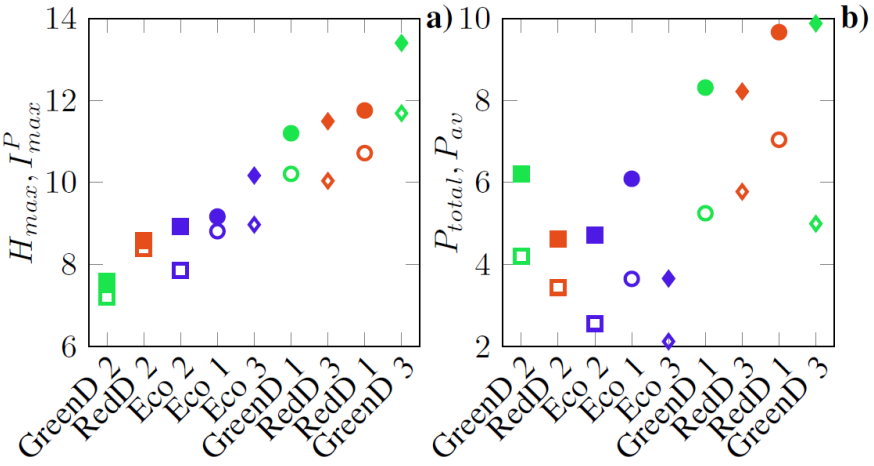

Figure 5.9: Plot a) shows maximal entropy (filled markers) and the related mutual information for all device sets. Plot b) shows P_{total} (filled) and the average of power states (hollow markers) from all devices.

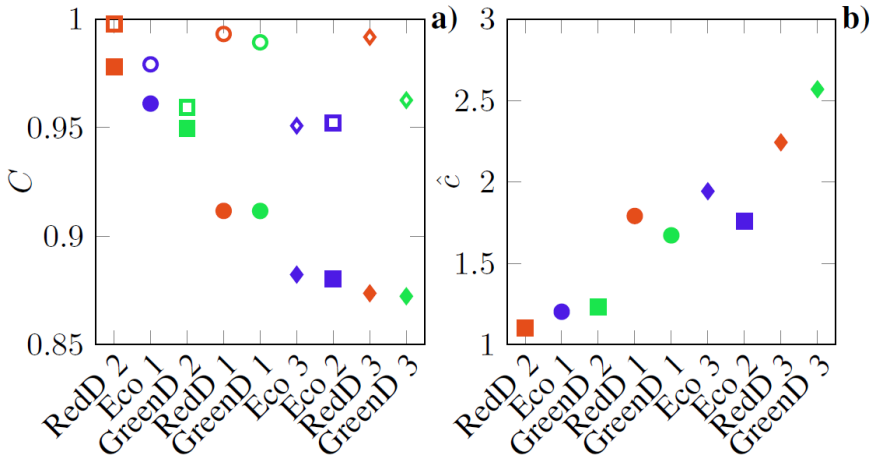

Figure 5.10: The proficiency, C, is shown in a). Filled markers show the maximal entropy case, hollow markers the values for $\hat{p} = 0.1$. Plot b) shows the average occupancy number, $\hat{c}$.

Plot a) of figure 5.10 shows the proficiency for the maximum entropy case (filled markers) and the proficiency for low device probability (hollow markers) of $\hat{p} = 0.1$. The latter are a sample of figure 5.12, which shows the influence of device usage rates on proficiency. Plot b) shows the average occupation numbers, $\hat{c}$, of power states. In general, it increases with decreasing proficiency, as expected from equation 5.16. For the real device sets, all values are between 1 and 3, which is far below the those from the artificial sets A and B, as listed in tables 5.5 and 5.2. The average occupation number measures average equality of power values. The appliance set complexity from [Ega15b] is a measure of similarity of power values (without considering their likelihood), which

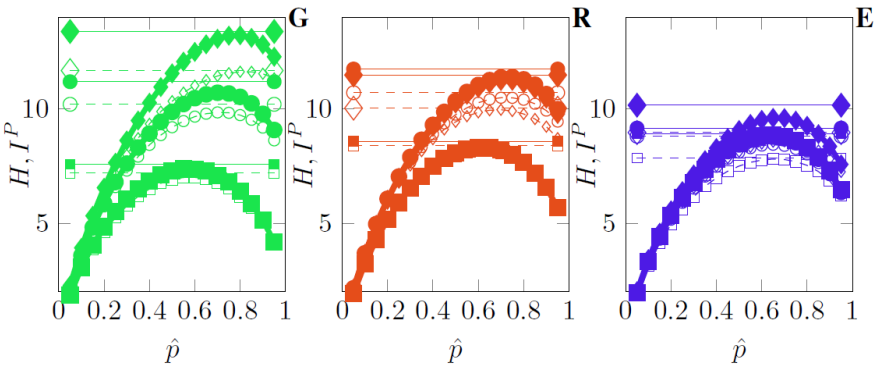

Figure 5.11: Entropy and mutual information are shown for all datasets. The horizontal lines mark the respective values for the maximum entropy case.

includes similarity, in a sense. If the distribution of modeling and measurement errors, which is assumed to be normal in [Ega15b], is of Delta type, the appliance set complexity is expected to match the value of $\hat{c}$ for a specific device set. Values for appliance set complexity are therefore always above $\hat{c}$.

As demonstrated in section 5.4, entropy and proficiency are a function of device operation probability. Figure 5.11 shows entropy and mutual information, in dashed lines, for each device set grouped by the three datasets: GreenD, RedD and Eco. The maximal entropy values are depicted by horizontal lines. Due to the multiple, differently distributed power values the parabolic curves reach the maximum above $\hat{p} = 0.5$ and meet the horizontal lines. The values of the selected Eco houses are more similar to each other than those of the GreenD houses, with the RedD houses in between.

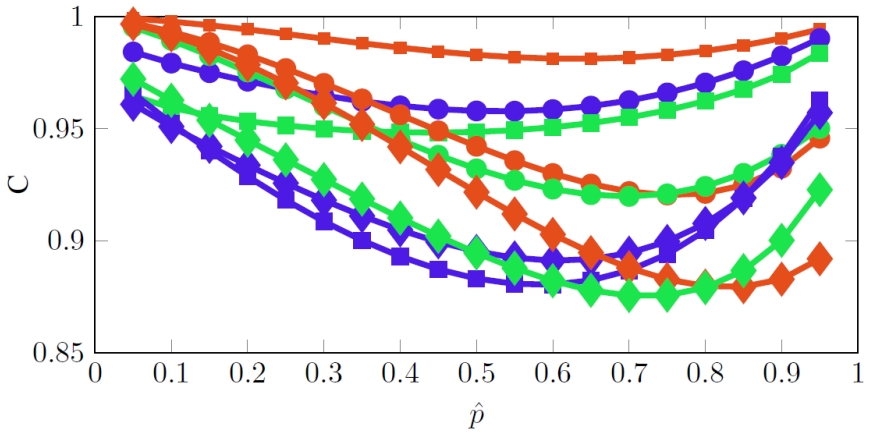

Figure 5.12: The proficiency, C, changes with the averaged device probability. The function is mainly related to the distribution of power values among the devices.

Figure 5.12 plots the proficiency as a function of average device operation probability, i. e., a processing of the values presented in figure 5.11. The device sets react differently to

varying device probability. The three RedD sets approach 1 when $\hat{p}$ is low, which means that the power values with few involved devices are generally more easily distinguishable. The device sets RedD3 and GreenD3 have their lowest C values at comparatively high $\hat{p}$, which means the indistinguishable power values include many devices, making them less likely to occur. The GreenD2 set is special in the sense that as proficiency is influenced by $\hat{p}$ very little, and it is ranked lowest according to the number of states, i. e., M or H_{max} in figure 5.10a, whereas for low $\hat{p} < 0.1$, proficiency is less than that of GreenD3, which is uppermost in figure 5.10a.

5.6 Discussion and Future Work

Load disaggregation can be identified as the decoding process within an information communication problem. The desired information is unintentionally coded in the power profile and the code, the mapping of device states to power values, depends exclusively on device attributes. The code cannot be modified, but a detailed understanding helps with decoding and enables the difficulty of decoding of a specific device set to be judged.

Entropy, as a measure of the amount of possible states (equaling possible device configurations), has the advantage that it adds up in the case of two merging device sets. This is not generally true for the mutual information of power values, which is an entropy-type measure also. The values for the maximal entropy case are a bound for more realistic cases that include the probabilities of devices to running and/or the power values. Proficiency can be seen as an estimate of the fraction of information about the device states which can be reproduced from the power values. Whether proficiency qualifies as an upper bound for detection rates of NILM algorithms cannot be said so far. In order to be meaningful, the proficiency calculation must consider the same attributes as utilized for disaggregation, which requires further development of the presented formalism to include other device parameters. Further, it is beneficial to define mutual information for continuous power values with respect to the signal to noise ratio and measurement accuracy.

The processing of power measurements on the device level allows the exploitation of the presented concepts more deeply. In addition to the single-device power values, the operation probabilities could be estimated. To identify characteristic parameters that are easily accessible could be of great practical value. For instance, knowing the relation between total consumed average power and the total power of the device set could allow an estimation of the average device runtimes. The fraction of time steps without any running devices allows similar reasoning.

The concepts presented in this chapter can be extended to any parameter space with other attributes, in [Yu-14]. Those can, but do not necessarily need to include power values. These concepts can further help to decide which attributes are more promising to distinguish specific scenarios for a device set. On the other hand single devices that can cause difficulties could be identified and potential solutions developed.

CHAPTER 6 Summary

"All things are difficult before they are easy."

– Thomas Fuller

Each of the main chapters, 3, 4 and 5, of this thesis, contain their own conclusions and outlines for the future. This section offers a brief summary and concludes the thesis, reflecting in particular on the problems stated in section 1.3. Further, the section contains a summary of a previously published piece of work related to the topic of complexity for load disaggregation.

6.1 Smart Microgrid Simulation

Simulation software for smart power systems is under development. Compared to traditional power grid simulation software requires additional features for the simulation of renewable sources and communication infrastructures. The vendors of commercial tools advance their product development accordingly. The situation is different for open-source software projects, as resources are more limited and commitment of volunteer developers is for a limited time. Combining existing tools for simulation of power systems and renewable sources is a first step towards a comprehensive smart grid simulator. More ambitious projects go beyond that, for instance, GridLAB-D is a project developed from scratch, dedicated to distribution system simulation. Another example is Mosaik, which follows a different approach, where a co-simulation platform is developed that allows the combining of existing software and addition of different parts if required. In addition to renewable sources, Mosaik requires residential loads to be modeled and simulated. Traditional approaches based on statistical data are insufficient for small-scale systems such as MGs. A single tool covering all aspects of smart MG simulation is not available freely. However, it is questionable whether this is needed at all. Many different effects that are the focus of simulation happen on broad range of timescales and therefore combining them in a single software tool has little benefit. Aside from simulation, there

exist a number of conceptual models that help to generate a common understanding. Examples of this would be SGAM, which focuses on interoperability, or RASSA, which focuses on secure and safe smart grid operation.

6.2 The RAPSim Simulation Tool

Renewable Alternative Power Systems Simulation is a software tool under development that aims to provide an easy-to-use smart microgrid simulation tool. In addition different representations of the grid physics, it contains models for renewable sources and residential loads. The software has been developed in Java under an open software license. It runs as a standalone application on different operating systems or can be executed from the user's JDE. Users can modify the code or implement new models for the objects in the smart grid. Such models are automatically included in the GUI. The application of RAPSim is time series simulation down to a one-minute resolution. RAPSim may be helpful for making engineering decisions within smart grids. Another application could be the development of microgrids' central controllers. Even though this is possible currently it could be facilitated by further adoption of the software structure. The replay of measured time series as a model (as for PV generation or residential consumption) is an existing feature that ought to be enhanced. Climate and weather data that are accessible for all the simulated objects is beneficial. The current weather simulation requires more detailed evaluation and can be enriched by playback of measured data for a specific location.

6.3 Proficiency of Power Values

Load disaggregation was modeled as a decoding process within an information communication problem. The description and improved understanding of the respective coding process can help in decoding. A general insight is that if only power values are processed with NILM, the coding scheme is unlikely to be entirely bijective, which means that not all possible device configurations are mapped to distinguishable power values. Such indistinguishable power values can present difficulties for NILM algorithms. The calculation of entropy of initial device states, mutual information of power values, and the resulting uncertainty coefficient or proficiency was shown. This demonstrated that the proficiency is highly dependent on the device's running probability, especially if devices with multiple values of power consumption are involved. Artificial device sets were used, as well as real measured values of devices that were repeatedly used for other load disaggregation studies to demonstrate the meaning of these parameters. The insights on the coding procedure from device states to aggregated power values contributes to the improvement of existing NILM algorithms.

Bibliography

[Alt14] H. Altrabalsi, J. Liao, L. Stankovic, and V. Stankovic. A low-complexity energy disaggregation method: Performance and robustness. In *Computational Intelligence Applications in Smart Grid (CIASG), 2014 IEEE Symposium on*, pages 1–8, Dec 2014.

[And13] K. Anderson, J. Du, A. Narayan, and A. E. Gamal. GridSpice: A distributed simulation platform for the smart grid. *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, 3203(c):1–5, 2013.

[Arm09] M. M. Armstrong, M. C. Swinton, H. Ribberink, I. Beausoleil-Morrison, and J. Millette. Synthetically derived profiles for representing occupant-driven electric loads in Canadian housing. *Journal of Building Performance Simulation*, 2(1):15–30, 2009.

[Bar04] M. Baranski and J. Voss. Genetic algorithm for pattern detection in NIALM systems. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 2004.

[Bec14] C. Beckel, W. Kleiminger, R. Cicchetti, T. Staake, and S. Santini. The ECO data set and the performance of non-intrusive load monitoring algorithms. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings - BuildSys '14*, pages 80–89, New York, New York, USA, nov 2014. ACM Press.

[Bie12] T. Bier, D. Abdeslam, J. Merckle, and D. Benyoucef. Smart meter systems detection and classification using artificial neural networks. In *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, pages 3324–3329, Oct 2012.

[Bin04] H. Bindner, O. Gehrke, P. Lundsager, J. C. Hansen, and T. Cronin. IPSYS–A simulation tool for performance assessment and controller development of integrated power system distributed renewable energy generated and storage. *WREC VIII, Denver, Colorado*, 2004.

[Bod15] T. Bodenbinder. *Das Architektur-Modell für Microgrids*. AV Akademikerverlag, Saarbrücken, 2015.

[Boe10] R. Boers, M. J. de Haij, W. M. F. Wauben, H. K. Baltink, L. H. van Ulft, M. Savenije, and C. N. Long. Optimized fractional cloudiness determination from five ground-based remote sensing techniques. *Journal of Geophysical Research*, 115(D24):D24116, dec 2010.

[Bom05] E. Bompard, E. Carpaneto, G. Ciwei, R. Napoli, M. Benini, M. Gallanti, and G. Migliavacca. A game theory simulator for assessing the performances of competitive electricity markets. In *2005 IEEE Russia Power Tech*, pages 1–9. IEEE, jun 2005.

[Bom08] E. Bompard and Y. Ma. Modeling Bilateral Electricity Markets: A Complex Network Approach. *IEEE Transactions on Power Systems*, 23(4):1590–1600, nov 2008.

[Can99] C. A. Canizares and F. Alvarado. UWPFLOW: continuation and direct methods to locate fold bifurcations in AC/DC/FACTS power systems. *University of Waterloo*, 1999.

[CEN12] CEN-CENELEC-ETSI Smart Grid Coordination Group. Smart Grid Reference Architecture. Technical Report November, CEN-CENELEC-ETSISmartGridCoordinationGrou, 2012.

[Ces13] R. Cespedes. Applications of a reference model to smarter electrical energy systems. *2013 IEEE Grenoble Conference*, pages 1–5, 2013.

[Cha08] D. P. Chassin, K. Schneider, C. Gerkensmeyer, S. Member, A. W. Gridlab-d, K. Schneider, C. Gerkensmeyer, and A. W. Gridlab-d. GridLAB-D: An Open-source Power Systems Modeling and Simulation Environment. In *2008 IEEE/PES Transmission and Distribution Conference and Exposition*, pages 1–5. IEEE, apr 2008.

[Cho09] S. Chowdhury, S. Chowdhury, and P. Crossley. *Microgrids and Active Distribution Networks*. 2009.

[Col09] C. M. Colson and M. H. Nehrir. A Review of Challenges to Real-Time Power Management of Microgrids. *IEEE Power & Energy Society General Meeting*, 2009.

[Col11] S. Cole and R. Belmans. MatDyn, A New Matlab-Based Toolbox for Power System Dynamic Simulation. *IEEE Transactions on Power Systems*, 26(3):1129–1136, aug 2011.

[Con05] G. Conzelmann, G. Boyd, V. Koritarov, and T. Veselka. Multi-agent power market simulation using EMCAS. In *IEEE Power Engineering Society General Meeting, 2005*, pages 2829–2834 Vol. 3, 2005.

[Cov05] T. M. Cover and J. A. Thomas. *Elements of Information Theory Wiley Series in Telecommunications and Signal Processing: Amazon.de: Thomas M. Cover, Joy A. Thomas: Fremdsprachige Bücher*. Wiley, 2005.

[Don13] R. Dong, L. Ratliff, H. Ohlsson, and S. S. Sastry. Fundamental Limits of Nonintrusive Load Monitoring. oct 2013.

[Dug11] R. C. Dugan and T. E. McDermott. An Open Source Platform for Collaborating on Smart Grid Research. In *Power and Energy Society General Meeting, 2011 IEEE*, number Ivvc, pages 1–7. IEEE, jul 2011.

[Ega15a] D. Egarter, V. Bhuvana, and W. Elmenreich. PALDi: Online Load Disaggregation via Particle Filtering. *IEEE Transactions on Instrumentation and Measurement*, 64(2):467–477, Feb 2015.

[Ega15b] D. Egarter, M. Pöchacker, and W. Elmenreich. Complexity of Power Draws for Load Disaggregation. Jan. 2015.

[EH98] S. H. El-Hefnawi. Photovoltaic diesel-generator hybrid power system sizing. *Renewable Energy*, 13(1):33–40, 1998.

[Elm08] W. Elmenreich and H. de Meer. Self-Organizing Networked Systems for Technical Applications: A Discussion on Open Issues. In J. Sterbenz and K. Hummel, Editors, *Proceedings of the Third International Workshop on Self-Organizing Systems*, pages 1–9. Springer Verlag, 2008.

[Fan11] X. Fang, S. Misra, G. Xue, and D. Yang. Smart Grid – The New and Improved Power Grid :. *IEEE Communications Surveys & Tutorials*, pages 1–37, 2011.

[Far12] M. O. Faruque, V. Dinavahi, M. Steurer, a. Monti, K. Strunz, J. a. Martinez, G. W. Chang, J. Jatskevich, R. Iravani, and a. Davoudi. Interfacing Issues in Multi-Domain Simulation Tools. *IEEE Transactions on Power Delivery*, 27(1):439–448, jan 2012.

[Fig13] M. Figueiredo, B. Ribeiro, and A. de Almeida. Electrical Signal Source Separation Via Nonnegative Tensor Factorization Using On Site Measurements in a Smart Home. *IEEE Transactions on Instrumentation and Measurement*, PP(99):1–1, 2013.

[Gon11a] H. Goncalves, A. Ocneanu, and M. Berges. Unsupervised disaggregation of appliances using aggregated consumption data. In *Proceedings of KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, 2011.

[Gon11b] H. Gonçalves, A. Ocneanu, M. Bergés, and R. Fan. Unsupervised disaggregation of appliances using aggregated consumption data. *The 1st KDD Workshop on Data Mining Applications in Sustainability (SustKDD)*, 2011.

[Got11] S. Gottwalt, W. Ketter, C. Block, J. Collins, and C. Weinhardt. Demand side management - A simulation of household behavior under variable prices. *Energy Policy*, 39(12):8163–8174, dec 2011.

[Gün11] V. C. Güngör, D. Sahin, T. Kocak, S. Ergüt, C. B. Buccella, C. Cecati, and G. P. Hancke. Smart Grid Technologies : Communication Technologies and Standards. *IEEE Transactions on Industrial Informatics*, 7(4):529–539, 2011.

[Häg10] U. Häger, A. Seack, C. Rehtanz, S. Lehnhoff, T. Zimmermann, and H. F. Wedde. Applicability of coordinated power flow control based on multi-agent systems. In *2010 IREP Symposium Bulk Power System Dynamics and Control - VIII (IREP)*, pages 1–8. IEEE, aug 2010.

[Har92] G. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[Ili07] M. D. Ilic. From Hierarchical to Open Access Electric Power Systems. *Proceedings of the IEEE*, 95(5):1060–1084, may 2007.

[Ili08] M. D. Ilic, U. A. Khan, and M. D. Ili. Modeling future cyber-physical energy systems. In *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century*, pages 1–9. IEEE, jul 2008.

[Iwa81] S. Iwamoto and Y. Tamura. A Load Flow Calculation Method for Ill-Conditioned Power Systems. *IEEE Transactions on Power Apparatus and Systems*, PAS-100(4):1736–1743, apr 1981.

[Joh13] M. J. Johnson and A. S. Willsky. Bayesian Nonparametric Hidden semi-Markov Models. *J. Mach. Learn. Res.*, 14(1):673–701, Feb. 2013.

[Kal04] J. Kaldellis. Optimum technoeconomic energy autonomous photovoltaic solution for remote consumers throughout Greece. *Energy Conversion and Management*, 45(17):2745–2760, 2004.

[Kha12a] T. Khatib, A. Mohamed, and K. Sopian. A review of solar energy modeling techniques. *Renewable and Sustainable Energy Reviews*, 16(5):2864–2869, 2012.

[Kha12b] T. Khatib, A. Mohamed, and K. Sopian. A Software Tool for Optimal Sizing of PV Systems in Malaysia. *Modelling and Simulation in Engineering*, 2012:1–11, 2012.

[Kim11] H. Kim, M. Marwah, M. Arlitt, G. Lyon, and J. Han. Unsupervised Disaggregation of Low Frequency Power Measurements. *SDM*, 2011.

[Kol11] J. Z. Kolter and M. J. Johnson. REDD: A Public Data Set for Energy Disaggregation Research. In *Proceeding of the SustKDD Workshop on Data Mining Applications in Sustainability*, 2011.

[Kol12] Z. Kolter and T. Jaakkola. Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation. In *Proceedings of the International Conference on Artifical Intelligence and Statistics*, 2012.

[Kra12] O. Kramer, O. Wilken, P. Beenken, A. Hein, A. HÌwel, T. Klingenberg, C. Meinecke, T. Raabe, and M. Sonnenschein. On Ensemble Classifiers for Nonintrusive Appliance Load Monitoring. In E. Corchado, V. Snášel, A. Abraham, M. Woźniak, M. Graña, and S.-B. Cho, Editors, *Hybrid Artificial Intelligent Systems*, volume 7208 of *Lecture Notes in Computer Science*, pages 322–331. Springer Berlin Heidelberg, 2012.

[Kru80] G. Krumpholz, K. Clements, and P. Davis. Power System Observability: A Practical Algorithm Using Network Topology. *IEEE Transactions on Power Apparatus and Systems*, PAS-99(4):1534–1542, jul 1980.

[Lal10] M. Lalwani, D. Kothari, and M. Singh. Investigation of Solar Photovoltaic Simulation Softwares. *International Journal of Applied Engineering Research*, 1(3):585–601, 2010.

[Las01] B. Lasseter. Microgrids [distributed power generation]. In *2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.01CH37194)*, volume 1, pages 146–149. IEEE, 2001.

[Las07] R. H. Lasseter. Certs Microgrid. *2007 IEEE International Conference on System of Systems Engineering*, pages 1–5, apr 2007.

[Li09a] H. Li and L. Tesfatsion. The AMES wholesale power market test bed: A computational laboratory for research, teaching, and training. In *2009 IEEE Power Energy Society General Meeting*, pages 1–8, 2009.

[Li09b] H. Li and L. Tesfatsion. Development of Open Source Software for Power Market Research: The AMES Test Bed. Staff general research papers archive, Iowa State University, Department of Economics, 2009.

[Lia10] J. Liang, S. Ng, G. Kendall, and J. Cheng. Load Signature Study, Part I: Basic Concept, Structure, and Methodology. *IEEE Trans. Power Del.*, 25(2):551–560, 2010.

[Lia14] J. Liao, G. Elafoudi, L. Stankovic, and V. Stankovic. Non-intrusive appliance load monitoring using low-resolution smart meter data. In *Proc. IEEE International Conference on Smart Grid Communications (SmartGridComm'14)*, Venice, Italy, 2014.

[Lid11] N. Lidula and A. Rajapakse. Microgrids research: A review of experimental microgrids and test systems. *Renewable and Sustainable Energy Reviews*, 15(1):186–202, jan 2011.

[Mah09] J. Mahseredjian, V. Dinavahi, and J. A. Martinez. Simulation Tools for Electromagnetic Transients in Power Systems : Overview and Challenges. *Transactions on Power Delivery*, 24(3):1657–1669, 2009.

[Mar10] C. Marinescu, a. Deaconu, E. Ciurea, and D. Marinescu. From Microgrids to Smart Grids: Modeling and simulating using graphs. Part I active power flow. *2010 12th International Conference on Optimization of Electrical and Electronic Equipment*, pages 1245–1250, may 2010.

[Mar11a] J. A. Martinez, V. Dinavahi, M. H. Nehrir, and X. Guillaud. Tools for Analysis and Design of Distributed Resources — Part IV: Future Trends. *IEEE Transactions on Power Delivery*, 26(3):1671–1680, 2011.

[Mar11b] J. A. Martinez and J. Martin-Arnedo. Tools for Analysis and Design of Distributed Resources — Part I: Tools for Feasibility Studies. *IEEE Transactions on Power Delivery*, 26(3):1643–1652, 2011.

[Mar13] L. Mariam, M. Basu, and M. F. Conlon. A Review of Existing Microgrid Architectures. *Journal of Engineering*, 2013:1–8, 2013.

[Mei11] M. Meisel, T. Leber, M. Ornetzeder, M. Stachura, A. Schiffleitner, G. Kienesberger, J. Wenninger, and F. Kupzog. Smart demand response scenarios. In *IEEE AFRICON 2011*, number September, pages 1–6, Livingstone, sep 2011. IEEE.

[Mil05] F. Milano. An Open Source Power System Analysis Toolbox. *IEEE Transactions on Power Systems*, 20(3):1199–1206, aug 2005.

[Mil09] F. Milano, M. Zhou, and G. Hou. Open model for exchanging power system data. In *2009 IEEE Power & Energy Society General Meeting*, pages 1–7. IEEE, jul 2009.

[Mil10] F. Milano. *Power system modelling and scripting*, volume 54. Springer, 2010.

[Mon85] A. Monticelli and F. Wu. Network Observability: Identification of Observable Islands and Measurement Placement. *IEEE Transactions on Power Apparatus and Systems*, PAS-104(5):1035–1041, may 1985.

[Mon13] A. Monacchi, W. Elmenreich, S. D. Alessandro, and A. M. Tonello. Strategies for Domestic Energy Conservation in Carinthia and Friuli-Venezia Giulia. In *Proceedings of the 39th Annual Conference of the IEEE Industrial Electronics Society*, 2013.

[Mon14] A. Monacchi, D. Egarter, W. Elmenreich, S. D'Alessandro, and A. M. Tonello. GREEND: an energy consumption dataset of households in Italy and Austria. In *Proc. of IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Venice, Italy, Nov 2014.

[Mon16] A. Monacchi, S. Zhevzhyk, and W. Elmenreich. HEMS: a home energy market simulator. *Computer Science - Research and Development*, 31(3):111–118, 2016.

[New10] M. Newman. *Networks: An Introduction*. Oxford University Press, USA, 2010.

[Nut07] J. Nutaro, P. T. Kuruganti, L. Miller, S. Mullen, and M. Shankar. Integrated Hybrid-Simulation of Electric Power and Communications Systems. In *2007 IEEE Power Engineering Society General Meeting*, pages 1–8. IEEE, jun 2007.

[Oli12] P. Oliveira, T. Pinto, H. Morais, and Z. Vale. MASGriP - A Multi-Agent Smart Grid Simulation Platform, 2012.

[Pae13] A.-g. Paetz, T. Kaschub, P. Jochem, and W. Fichtner. Load-shifting potentials in households including electric mobility - A comparison of user behaviour with modelling results. In *2013 10th International Conference on the European Energy Market (EEM)*, pages 1–7. IEEE, may 2013.

[Pal14] P. Palensky, E. Widl, A. Elsheikh, and S. Member. Simulating Cyber-Physical Energy Systems: Challenges, Tools and Methods. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(3):318–326, mar 2014.

[Par14] O. Parson, S. Ghosh, M. Weal, and A. Rogers. An unsupervised training method for non-intrusive appliance load monitoring. *Artificial Intelligence*, 217(0):1 – 19, 2014.

[Pat12] S. Pattem. Unsupervised Disaggregation for Non-intrusive Load Monitoring. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 515–520, Dec 2012.

[Per10] L. Peretto. The role of measurements in the smart grid era. *IEEE Instrumentation and Measurement Magazine*, 13(June):22–25, 2010.

[Pip09] M. Pipattanasomporn, H. Feroze, and S. Rahman. Multi-agent systems in a distributed smart grid: Design and implementation. *2009 IEEE/PES Power Systems Conference and Exposition*, pages 1–8, mar 2009.

[Pöc13] M. Pöchacker, A. Sobe, and W. Elmenreich. Simulating the smart grid. In *2013 IEEE Grenoble Conference*, pages 1–6, Grenoble, jun 2013. IEEE.

[Pöc14] M. Pöchacker, T. Khatib, and W. Elmenreich. The Microgrid Simulation Tool RAPSim: Description and Case Study. In *IEEE Proceedings of ISGT Asia 2014*, pages 287–292, Kuala Lumpur, may 2014. IEEE.

[Pöc15] M. Pöchacker and W. Elmenreich. Model implementation for the extendable open source power system simulator RAPSim. In *Intelligent Solutions in Embedded Systems (WISES), 2015 12th International Workshop on*, pages 103–108, 2015.

[Pöc16] M. Pöchacker, D. Egarter, and W. Elmenreich. Proficiency of Power Values for Load Disaggregation. *IEEE Transactions on Instrumentation and Measurement*, 65(1):46–55, 2016.

[Pur05] K. Purchala, L. Meeus, D. Van Dommelen, and R. Belmans. Usefulness of DC Power Flow for Active Power Flow Analysis. In *IEEE Power Engineering Society General Meeting, 2005*, pages 2457–2462. IEEE, 2005.

[Ran10] V. V. Ranade and J. Beal. Distributed Control for Small Customer Energy Demand Management. *2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, pages 11–20, sep 2010.

[SB06] R. Soler-Bientz, L. Ricalde-Cab, and L. Solis-Rodriguez. Developing a mobile stand alone photovoltaic generator. *Energy Conversion and Management*, 47(18):2948–2960, 2006.

[Sch09] K. P. Schneider, D. Chassin, Y. Chen, and J. C. Fuller. Distribution power flow for smart grid technologies. In *2009 IEEE/PES Power Systems Conference and Exposition*, pages 1–7. IEEE, mar 2009.

[Sch11] S. Schütte, S. Scherfke, and M. Tröschel. Mosaik: A framework for modular simulation of active components in Smart Grids. In *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*, pages 55–60. IEEE, oct 2011.

[Sch12] S. Schütte, S. Scherfke, and M. Sonnenschein. Mosaik - Smart Grid Simulation API - Toward a Semantic based Standard for Interchanging Smart Grid Simulations. In *SMARTGREENS*, pages 14–24, 2012.

[Sha08] S. Shaw, S. Leeb, L. Norford, and R. Cox. Nonintrusive Load Monitoring and Diagnostics in Power Systems. *IEEE Trans. Instrum. Meas.*, 57(7):1445–1454, July 2008.

[Sha13] H. Shao, M. Marwah, and N. Ramakrishnan. A Temporal Motif Mining Approach to Unsupervised Energy Disaggregation: Applications to Residential and Commercial Buildings. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA.*, 2013.

[Sob12] A. Sobe and W. Elmenreich. Smart Microgrids: Overview and Outlook. In *Proceedings of the GI INFORMATIK Workshop on Smart Grids*, number 1, Braunschweig, 2012.

[Sri06] D. Srinivasan, W. S. Ng, and A. Liew. Neural-network-based signature recognition for harmonic source identification. *IEEE Trans. Power Del.*, 21(1):398–405, 2006.

[Tin67] W. F. Tinney and C. E. Hart. Power Flow Solution by Newton's Method. *IEEE Transactions on Power Apparatus and Systems*, PAS-86(11):1449–1460, nov 1967.

[Tre12] J. Trefke, C. Danekas, S. Rohjans, and J. M. Gonzalez Vazquez. Adaptive architecture development for Smart Grids based on integrated building blocks. In *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–8. IEEE, oct 2012.

[Übe14] S. Übermasser, C. Niklas, R. Braun, M. Meisel, and T. Leber. GeoGreen - Optimizing green energy and grid load by geographical steering of energy consumption, 2014.

[Van07] L. Vanfretti and F. Milano. Application of the PSAT, an Open Source Software, for Educational and Research Purposes. *2007 IEEE Power Engineering Society General Meeting*, pages 1–7, jun 2007.

[Vyt10] P. Vytelingum, S. D. Ramchurn, T. D. Voice, A. Rogers, and N. R. Jennings. Trading agents for the smart electricity grid. *9th International Conference on AAMAS 2010*, pages 897–904, 2010.

[Wau10] W. Wauben, M. de Haij, R. Boers, H. Klein Baltink, B. van Ulft, and M. Savenije. On the Generation of an Optimized Fractional Cloudiness Time Series using a Multi-Sensor Approach. Technical report, Royal Nethrlands Meteorological Institute (KNMI), De Bilt, 2010.

[Whi08] J. V. White, S. Steingold, and C. G. Fournelle. Performance Metrics for Group-Detection Algorithms Mathematical formulation. In *Computing Science and Statistics*, page 15, 2008.

[Wid12] E. Widl, P. Palensky, and A. Elsheikh. Evaluation of two approaches for simulating cyber-physical energy systems. *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, pages 3582–3587, oct 2012.

[Wij13] T. K. Wijaya, D. Banerjee, T. Ganu, D. Chakraborty, S. Battacharya, T. Papaioannou, D. P. Seetharam, and K. Aberer. DRSim: A cyber physical simulator for Demand Response systems. In *2013 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 217–222. IEEE, oct 2013.

[Xu15] Y. Xu and J. Milanovic. Artificial-Intelligence-Based Methodology for Load Disaggregation at Bulk Supply Point. *IEEE Transactions on Power Systems*, 30(2):795–803, March 2015.

[Yu-14] Yu-Hsiu Lin and Men-Shen Tsai. Development of an Improved Time–Frequency Analysis-Based Nonintrusive Load Monitor for Load Demand Identification. *IEEE Transactions on Instrumentation and Measurement*, 63(6):1470–1483, jun 2014.

[Zei11] M. Zeifman and K. Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE Trans. Consum. Electron.*, 57(1):76 –84, February 2011.

[Zha14] F. Zhang, Jian (IREENA, I'Univ. de Nantes, St. Nazaire, P.-E. Guillerm, L. Moreau, and M. Machmoum. State of the art in tidal current energy extracting technologies. In *2014 First International Conference on Green Energy ICGE 2014*, page 7, Sfax, 2014. IEEE.

[Zho07] M. Zhou and S. Zhou. Internet, Open-source and Power System Simulation. In *2007 IEEE Power Engineering Society General Meeting*, pages 1–5. IEEE, jun 2007.

[Zia11] T. Zia, D. Bruckner, and A. Zaidi. A hidden Markov model based procedure for identifying household electric loads. In *Proceedings of Annual Conference on IEEE Industrial Electronics Society (IECON)*, 2011.

[Zim09] R. D. Zimmerman, C. E. Murillo-Sanchez, R. J. Thomas, C. E. Murillo-s, and R. J. Thomas. MATPOWER's extensible optimal power flow architecture. In *2009 IEEE Power & Energy Society General Meeting*, number x, pages 1–7. IEEE, jul 2009.

[Zim10] R. D. Zimmerman. Uniform Price Auctions and Optimal Power Flow. 2010.

[Zim11a] R. D. Zimmerman. Matpower User´s Manual. 2011.

[Zim11b] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas. MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education. *IEEE Transactions on Power Systems*, 26(1):12–19, feb 2011.

[Zoh12] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar. Non-Intrusive Load Monitoring Approaches for Disaggregated Energy Sensing: A Survey. *Sensors*, 12(12):16838–16866, 2012.

[Zoh13] A. Zoha, A. Gluhak, M. Nati, and M. Imran. Low-power appliance monitoring using Factorial Hidden Markov Models. In *Proceedings of IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2013.