*energies*

MDPI

*Article*

# Methodology for the Automatic Generation of Optimization Models of Systems of Flexible Energy Resources

**Lukas Peter Wagner [1,*], Felix Gehlhoff [1], Lasse Matthias Reinpold [1], Georg Frey [2], Julian Jepsen [3,4] and Alexander Fay [5,*]**

1 Institute of Automation Technology, Helmut Schmidt University, 22043 Hamburg, Germany; felix.gehlhoff@hsu-hh.de (F.G.); lasse.reinpold@hsu-hh.de (L.M.R.);
2 Chair of Automation and Energy Systems, Saarland University, 66123 Saarbrücken, Germany; georg.frey@aut.uni-saarland.de
3 Institute of Materials Science, Helmut Schmidt University, 22043 Hamburg, Germany; jepsen@hsu-hh.de
4 Institute of Hydrogen Technology, Helmholtz-Zentrum Hereon, 21502 Geesthacht, Germany
5 Chair of Automation, Ruhr University, 44801 Bochum, Germany
* Correspondence: lukas.wagner@hsu-hh.de (L.P.W.); alexander.fay@ruhr-uni-bochum.de (A.F.)

**Abstract:** The integration of increasing shares of intermittent renewable energy necessitates flexibility in both energy generation and consumption. Typically, the operation of flexible energy resources is orchestrated through optimization models. However, the manual creation of these models is a complex and error-prone task, often requiring the expertise of domain specialists. This work introduces a methodology for the automatic generation of optimization models for systems of flexible energy resources to simplify the modeling process and increase the use of energy flexibility. This methodology utilizes a modular, generic model structure designed to depict systems of flexible energy resources. It incorporates algorithms for model parameter derivation from operational data and an information model that represents the system's structure and dependencies of resources. The efficacy of this methodology is demonstrated in two case studies, highlighting its relevance and ability to significantly streamline the optimization modeling process by minimizing the need for manual intervention.

**Keywords:** automatic model generation; energy flexibility; optimization model



Academic Editor: Adrian Ilinca

Received: 4 December 2024
Revised: 6 January 2025
Accepted: 9 January 2025
Published: 13 January 2025

**Citation:** Wagner, L.P.; Gehlhoff, F.; Reinpold, L.M.; Frey, G.; Jepsen, J.; Fay, A. Methodology for the Automatic Generation of Optimization Models of Systems of Flexible Energy Resources. *Energies* **2025**, *18*, 325. https://doi.org/10.3390/en18020325



## 1. Introduction

The energy transition aims at the large-scale electrification of the energy system to achieve decarbonization. This requires an increasing share of energy generation from renewable sources, which in turn demands flexibility in both generation and consumption to address the challenges of intermittent energy supply [1]. The term *energy flexibility* refers to the ability of a resource to modulate its power input or output [2]. Therein, it must be ensured that the modulation of the power input or output of any resource is conducted such that resource operators' targets are met. A target can be an energy production or consumption target or targets for the volume of produced goods [3]. Resource operators are those who can directly influence the operation of flexible energy resources [4]. A study by Misconel et al. [5] finds that the additional provisioning of energy flexibility of up to 12 GW is needed in the German electricity zone by 2030 to maintain the balance between generation and demand. Various studies also show that providing energy flexibility, such as ancillary services [6] or by adapting schedules to market prices [7], holds significant financial potential for system operators [6,7]. Furthermore, the German Federal

Network Agency introduced plans on updating regulations regarding dynamic network charges to encourage system operators to adjust their energy consumption according to grid demands [8]. These plans further increase the economic value of energy flexibility.

A systematic review of modeling approaches for flexible energy resources conducted by Wagner et al. [9] finds that flexible energy resources are usually not operated independently, but rather in systems with other flexible energy resources, such as electrolyzers, combined heat and power plants, or battery energy storage systems. This integration enhances the overall flexibility potential of the system, surpassing that of the individual resources involved [10,11]. This review [9] also finds that the majority of models applied for the operational planning of systems of flexible energy resources are mixed-integer linear programming (MILP) optimization models. A comparison of (mixed-integer) linear and nonlinear optimization models has shown that (mixed-integer) linear optimization models are preferred for generating optimized schedules for systems of energy resources due to their comparatively short computational times [12] while achieving near globally optimal solutions with efficient heuristics [13]. However, purely linear models are inadequate because they fail to capture non-linear resource characteristics, such as discrete operational states (e.g., "on", "off", ...), which often need to be incorporated into a model for the operational optimization of flexible energy resources [14].

Modeling the operational behavior of resources is key for planning their energy flexible operation. However, an analysis of "obstacles" impeding the use of energy flexibility conducted by Leinauer et al. [15] finds that there are a number of "*competence obstacles*". Key among these are the insufficient "*internal resources to deal intensively with [demand response] projects*" (In contrast to the use of the term *resources* for parts of a technical system throughout this work, this quote uses *resources* in the context of *human resources*) and a deficiency in employee skills "*to implement [demand response] measures*" [15]. Similarly, Krishnamoorthy and Skogestad [16] state that "*developing good [...] models is often challenging and expensive*". In line with the obstacles and challenges, Allen et al. [17] state that in general, "*[t]he successful construction of a model requires combining expertise in the domain being modeled with expertise in the practice of building models*". Furthermore, manual modeling could lead to inaccurate models [18]. The use of inaccurate models for the generation of optimized schedules for the control of flexible energy resources leads to deviations of optimized and realized schedules and poses problems to resource operators, e.g., in failing to achieve production and/or energy conversion targets.

To simplify and increase the use of energy flexibility and to ensure accurate modeling of the resource behavior, this work investigates how to automate the generation of optimization models of systems of flexible energy resources. The focus lies on developing a model generation methodology that minimizes manual involvement and eliminates the need for domain-specific knowledge about optimization modeling of flexible energy resources. To achieve this research objective, the following requirements (#) must be met:

(1) A **generic optimization model structure** suitable for the representation of a system of arbitrary (flexible) energy resources and its operational optimization is necessary [3,19–21]. The model structure must be able to capture all necessary flexibility features identified by Wagner et al. [3], such as operational boundaries, input–output relationships, energy balances for storage systems, system states, including state sequences and holding durations, ramp limits, as well as connections between resources [3].

(2) To identify suitable parameters for instances of the generic model structure, it is necessary to apply a **methodology for the automatic derivation of parameters** from time series data of the resources' operation and from an information model-based representation of the system's structure for the derivation of connections of

resources. This ensures that the parameters are reliable and that they fit the model structure (Requirement ①). In this context, a system's structure refers to the layout of system elements and their dependencies in the form of the flow of energy, material, and information [22,23].

③ A **methodology for the automatic generation of a parameterized optimization model instance** for a specific system consisting of resources is required. This methodology entails choosing the suitable elements of the model structure and their parameterization, utilizing the parameter set derived with the methodology that fulfills Requirement ② [21,24,25].

④ The **traceability** and **comprehensibility** of the resulting model must be ensured for the end-users of the methodology (Requirement ③), meaning that the eventual model composition is recognizable and the parameters have direct physical meanings. Additionally, suitable evaluation metrics must be used to convey the accuracy of the generated models [26–29].

This work is an extension of two previous works [3,23] already covering artifacts corresponding to Requirements ① and ②:

Requirement ① is already met by the generic model structure described by Wagner et al. [3]. Wagner et al. [3] conduct an analysis on necessary constraints "*to sufficiently model and optimize the operation of a flexible energy resource or a system thereof*". This research presented a reusable formulation of sets of MILP constraints as *flexibility features* [3].

The fulfillment of Requirement ② demands a methodology for the automatic derivation of parameters for optimization models. With this objective, the work by Wagner and Fay [23] investigates how "*to reduce the parameterization effort of a generic MILP optimization model (...) by means of automatic derivation of numerical parameters from time series data of the resource operation*". The work also investigates how information about connections of flexible energy resources can be extracted from an information model representing the system's structure [23].

Thus, to address the research objective of this work and the previously unaddressed Requirements ③ and ④, this work introduces a methodology for the automatic generation of optimization models for systems of flexible energy resources. Additionally, this work includes a comprehensive evaluation of the methodology, demonstrating its effectiveness and applicability in real-world scenarios as well as the models' traceability and comprehensibility by end-users.

The remainder of this work is structured as follows: Section 2 describes the analysis of related work as well as the respective fulfillment of the requirements. Section 3 explains the methodology for the automatic generation of optimization models. Section 4 describes the evaluation of the methodology via two case studies. Section 5 discusses the methodology and its evaluation. Section 6 concludes this work and outlines future work.

## 2. Related Work

This section analyzes related work in automatic model generation, particularly focusing on the fulfillment of Requirements ③ and ④. While previous work by the authors presented artifacts for the satisfaction of Requirements ① [3] and ② [23], analyzing the fulfillment of Requirements ① and ② in related work remains crucial to ensure their applicability in the context of optimization of systems of flexible energy resources. This analysis is structured into two main parts: the generation of *optimization* models (Section 2.1) and *simulation* models (Section 2.2). The analysis of related work is extended to the research area of the automatic generation of *simulation models* to include a wider variety of methodological approaches into the analysis. The analyses are summarized in Section 2.3.

### *2.1. Automatic Generation of Optimization Models*

Chicco and Mancarella [20] present an approach for automating the modeling of energy systems utilizing the *Energy Hub* concept, with a focus on optimizing operations within the context of an energy market. This concept accounts for the interactions among resources and external energy networks by using an input–output matrix that represents the respective conversion efficiencies of resources within the energy system. The automation of constructing the system's input–output matrix is realized by employing graph theory and backtracking techniques to explore the system's topology, represented by "interconnection matrices" for each of the resources. Nevertheless, the approach necessitates manual input in defining the conversion efficiencies of individual resources, which are crucial for the accurate representation and optimization of the system's operations, thereby not fully meeting Requirement ②. The model is nonlinear due to the multiplication of dispatch factors and power flows and captures only static input–output conversions. Consequently, off-design characteristics—operational behavior of the resource when it functions under conditions that deviate from those it was specifically engineered for—are not considered. Additionally, the modeling concept lacks the capability to depict system states. Conversion factors depending on the operating point, i.e., complex input–output relationships, and system states are, however, necessities according to Requirement ①. Thus, Requirement ① is only partially addressed. As the objective of the approach is the automation of model creation, akin to selecting model parts, but not its parameterization, Requirement ③ is partially met. In line with the fulfillment of Requirement ③, Requirement ④ is also partially met because the creation of the model is traceable. However, since no parameters are derived within this approach, their traceability cannot be evaluated.

Building upon the work by Chicco and Mancarella [20], Wang et al. [21] propose an automated and linearized modeling approach. The focus of their work lies on automating the modeling process for *Energy Hubs* by linearizing the input–output matrices, accomplished by eliminating 'dispatch factors' from the coupling matrices. The achievements of Wang et al. [21] in terms of requirement fulfillment are similar to those of the approach introduced by Chicco and Mancarella [20], as the efforts by Wang et al. [21] are directed towards simplifying the optimization process rather than enhancing the parameter derivation (Requirement ②) or incorporating accuracy-enhancing flexibility features like system states or more intricate input–output relationships (Requirement ①). Thus, Requirements ① and ③ are partially met, whereas Requirement ② remains unfulfilled. Requirement ④ is also partially met, as the creation of the matrices is traceable.

Henkel et al. [19] developed a method for the scheduling of modular electrolysis plants, a type of flexible energy resource, utilizing a multi-agent system. The scheduling agents' models are parameterized through the use of the *Module Type Package* (a standardized information model) and measurement data. The data are employed to calculate parameters for a quadratic approximation of the input–output relationships. The instantiation of agents for each electrolysis resource is achieved by analyzing system composition information. However, the objective of Henkel et al. [19] is the instantiation and parameterization of a multi-agent system but not the selection of necessary model components, thereby partially meeting Requirement ③. Each agent exhibits an identical model. Furthermore, the method does not include algorithms for the derivation of a complete set of parameters for the model. Hence, it does not fully satisfy Requirement ②. The decentralized optimization model developed by Henkel et al. [19] is specifically designed for the operational optimization of modular electrolysis plants and includes most of the flexibility features called for by Requirement ①, thereby partially fulfilling Requirement ①. The traceability and comprehensibility of the model is given due to is uniform structure and the customization to fit the parameters from the *Module Type Package*.

Kasper et al. [30] introduce a framework for the optimization of a resource used for the waste heat recovery of a steel production process. Therein, parts of an optimization model are parameterized based on measurement data, but no model parts are selected. Thus, Requirement ② is only partially satisfied. The application of the model used by Kasper et al. [30] is restricted to this specific resource type; hence, Requirement ① is also only partially met. Furthermore, Kasper et al. [30] specifically did not focus on creating a method for the automatic generation of optimization models (Requirement ③). As the model itself is based on physical principles as well as traceable parameters, the modeling process satisfies Requirement ④.

Förderer et al. [31] developed an approach for the representation of "feasible load profiles" of certain flexible energy resources (combined heat and power and battery energy storage systems) using artificial neural networks. In contrast to the commonly used approach of operational optimization [9], the work of Förderer et al. [31] proposes the rule-based use of these load profiles for the control of energy resources. However, this approach does not guarantee traceability nor comprehensibility of the model due to it being a black box model (Requirement ④) and does not allow for the explicit modeling of all necessary flexibility features, thus not meeting Requirement ①. As data are used for the modeling but no parameters with direct physical meaning and no information on resource connections are derived, Requirement ② is only partially met. As no optimization model is generated, Requirement ③ remains unsatisfied.

Manna et al. [32] present a data-driven optimization framework designed for the utilization of the flexibility of an evaporative cooling tower in response to prices from energy and reserve markets. Manna et al. [32] primarily focus on modeling bidding strategies for one specific resource type, i.e., cooling towers, thus only partially fulfilling Requirement ①. The method of Manna et al. [32] employs neural networks to model resource behavior, simplifying these nonlinear models into linear constraints for MILP optimization. Thus, they partially fulfill Requirement ② by using the neural network for deriving parameters. However, only one resource is considered instead of a system of resources. Requirement ③ is met through the transformation of the neural network's parameters into MILP constraints. Requirement ④ is not fulfilled due to the model structure and the parameter derivation approach resulting in a black box model.

### *2.2. Automatic Generation of Simulation Models*

Apart from the related work referenced in Section 2.1, there exist a number of works focusing on the automatic generation of *simulation* models. This section outlines works in this research area. The selected works provide combinations of fulfillments of requirements mostly not shown by the works analyzed in Section 2.1. None of the works analyzed in this section fulfill Requirement ①, as the models are simulation models.

Barth and Fay [24] investigated the automation of the generation of simulation models using existing engineering data. Therein, they proposed using an object-oriented data exchange format to facilitate the automatic generation of low-fidelity simulation models from piping and instrumentation diagrams [24]. This method leverages the simulation model library of *Modelica* for equation-based modeling of a system [24] but not for the modeling of flexible energy resources for optimization, hence not satisfying Requirement ①. The topology of the system is extracted from the object-oriented data exchange format, but additional parameters are manually extracted from other simulation applications [24], thus not fully meeting Requirement ②. As the generated model is not fully parameterized, Requirement ③ is only partially met. The generated model is traceable, as the model is created based on components present in the piping and instrumentation diagram. Thus, the approach meets Requirement ④.

The work of Ramonat and Fay [33] presents a concept for the automatic calibration and maintenance of simulation models in brownfield process plants. This includes parameter alignment, deviation detection, deviation cause detection, and model adaptation to ensure and maintain high fidelity of simulation models by continuously aligning model parameters with actual measurement data and adjusting for deviations due to plant changes or model inaccuracies. In subsequent work, Ramonat and Fay [22] describe a method for the direct or indirect parameterization of an existing simulation model based on measurement data. In the case of indirect parameterization, parameters must be converted using fluid dynamic formulas, whereas direct parameterization directly uses the measurement values. The application of their method requires manual effort in selecting the formulas and assigning the measured values. Therefore, the approach by Ramonat and Fay [22] partially meets the criteria of Requirement ②. Requirement ① is also not satisfied because Ramonat and Fay [33] suggest using a model library (*Modelica*) for the simulation of process plants, which is not suitable for representing systems of flexible energy resources. Since the focus is on calibration and maintenance [33], the process excludes automatic generation, leaving Requirement ③ unmet. Moreover, an extension to the method [33] by Ramonat et al. [26] highlights the importance of clearly explaining deviations through the use of an adaptive causal directed graph, thereby addressing part of Requirement ④. While it does not enhance the transparency of the model as a whole, it does provide explanations for parameter adjustments that are necessary to ensure alignment of the model with the system [26].

Martinez et al. [34] propose a methodology for automatically generating simulation-based digital twins of industrial process plants based on data from engineering sources, such as piping and instrumentation diagrams, technical data sheets, and control application programs. The work of Sierla et al. [35] builds upon the methodology of Martinez et al. [34] and outlines a seven-step semi-automatic methodology for creating digital twins from existing brownfield plant data. Information is also extracted from a piping and instrumentation diagram and converted into a graph. Then, a simulation model is created from this graph, requiring manual editing to finalize the model, configuring simulation model calculations and parameterizing the model based on recent process measurements [35]. The focus of the methodology presented by Sierla et al. [35] is on defining the process for graph preprocessing, model generation, and manual editing. The methodologies developed by Martinez et al. [34] and Sierla et al. [35] aim at reducing modeling efforts by automating the creation of simulation models, which partially fulfills Requirement ③. However, manual effort is still required for parameterization, indicating that these approaches do not completely eliminate the need for human intervention in the modeling process. Thus, Requirement ② is also only partially met, as parameters are neither fully nor automatically derived. As the model built is a simulation model for industrial process plants, Requirement ① is not met. The step-wise approach allows for the traceability of the model, thus meeting Requirement ④.

### *2.3. Summary of the Analysis of Related Work and Research Gaps*

The analysis of related work, summarized in Table 1, shows the fulfillment of Requirements ① to ④. The analyses have been split into the analysis of works in automatic generation of *optimization* and *simulation models*.

**Table 1.** Fulfillment of requirements for automatic model generation.

| Requirement | ① | ② | ③ | ④ |
|---|---|---|---|---|
| **Optimization Models** | | | | |
| Chicco and Mancarella [20] | ◐ | ◐ | ◐ | ◐ |
| Wang et al. [21] | ◐ | ◐ | ◐ | ◐ |
| Henkel et al. [19] | ◐ | ◐ | ◐ | ● |
| Kasper et al. [30] | ◐ | ◐ | ○ | ● |
| Förderer et al. [31] | ○ | ◐ | ○ | ○ |
| Manna et al. [32] | ◐ | ◐ | ● | ○ |
| **Simulation Models** | | | | |
| Barth and Fay [24] | ○ | ◐ | ◐ | ● |
| Ramonat et al. [22,26,33] | ○ | ◐ | ○ | ◐ |
| Martinez et al. [34], Sierla et al. [35] | ○ | ◐ | ◐ | ● |

Fulfillment: ● —full; ◐ —partial; ○ —none.

Some studies focus on *optimization models* for managing the operation of systems of energy resources [19–21,30–32]. These optimization models (Requirement ①) are often designed to represent specific resource types, such as modular electrolysis plants [19] or evaporative cooling towers [32], and thus are not generic in their structure. Further, they often lack the essential model components needed to depict flexibility features, including precise input–output relationships [20,21], system states [20,21], or ramp limits [19–21]. Methods for automatically deriving sets of parameters (Requirement ②) are occasionally described, but if so, only use-case specific methods with limited scope are presented [19,30]. This is also a key finding of the analysis of related work by Wagner and Fay [23]. Information on the system's structure is rarely used during parameter derivation. The automatic generation of a parameterized instance for a system of specific resources (Requirement ③) is often mentioned but seldom fully described [19–21,32].

However, the effectiveness of these methods [19–21,30] is limited by the deficiencies in the foundational optimization models and the incomplete methods for their parameterization, making them not universally applicable for the automatic generation of optimization models of systems of arbitrary flexible energy resources.

An analysis including approaches in automatically generating *simulation models* in Section 2.2 shows that standardized information models provide the foundation for the extraction of information on the system's structure, such as the work of Barth and Fay [24]. This information is then used for the selection of predefined model blocks, commonly accomplished in applications such as *Modelica* [24]. Several works introduce techniques for automatically creating *simulation* models that do not consider the energy flexibility of resources [24,33–35]. For high-fidelity models, full parameterization is executed [33] with parameters obtained from measurement data [22].

The traceability and comprehensibility of the generated models, particularly concerning the selection of necessary model components and the derivation of parameters, are fully addressed in only a few works [19,24,34,35]. Other studies provide approaches that are partially traceable, offering some level of transparency in their modeling processes [20,21,26,30].

The analyses show that there are works covering parts of the research objective; however, none of the works analyzed cover all the requirements. Thus, there remains a significant need for research into the automation of the generation of optimization models for systems comprising various flexible energy resources. Additionally, ensuring the trace-

ability and comprehensibility of the generated model is crucial for building trust among end-users [27]. When users can clearly understand how a model is constructed and follow the logic behind the parameter derivation, this significantly enhances their confidence in the results produced by the model. Transparent processes are key to facilitating broader adoption of energy flexibility and reliance on these models in practical applications, as they allow users to validate the underlying methodologies themselves.

## 3. Methodology for the Automatic Generation of Optimization Models

For enhanced clarity of the methodology, Section 3.1 outlines the process for the generation of optimization models. Next, the methodology for the automatic generation of optimization models is outlined. This methodology builds upon the results of the analysis in Section 2, which concludes that existing approaches for the generation of optimization models do not fully automate this process. Extending the analysis to *simulation models* reveals a promising approach for the automatic generation of an optimization model employing a generic model structure capable of accurately representing the system of resources under consideration (Section 3.2). This model structure is complemented by a methodology for the automatic derivation of necessary parameters (Section 3.3.1). The generic model is then utilized to automatically generate an optimization model instance for a specified system of resources (Section 3.3.2) using the parameters previously derived. The implementation of the methodology is described in Section 3.4.

### *3.1. Process for the Generation of Optimization Models*

This section describes the process for the generation of optimization models of systems of flexible energy resources as well as the necessary steps.

Figure 1 shows the process for the manual generation of an optimization model of a flexible energy resource for its control. This process can also be applied for multiple resources within a system. Preceding this process is the in-depth understanding of the operational behavior of the resource(s) to be modeled. This manual process requires domain knowledge in optimization modeling of energy resources.

The process for generating an optimization model from scratch, displayed in Figure 1, involves creating constraints (step 1), such as those for the representation of resource behavior, formulating an objective function (step 2), determining parameters (step 3), and utilizing these parameters for the parameterization of the constraints (step 4). Constraints and the objective function are functions of decision variables and parameters. In MILP modeling, decision variables can be multiplied with parameters but must not be multiplied with other decision variables. The parameterization of constraints also involves setting production targets and start parameters, such as initial system states or starting values for the state of charge (SOC) of (energy) storage systems. It is important to note that without specifying a target, such as a specific energy amount for the system output flow to be achieved during the optimization horizon, solving the optimization model instance may result in an 'empty' or at least a highly unsuitable schedule. This occurs because minimal costs can typically be achieved by not operating the resources within the system or, e.g., only in periods of negative electricity prices.

Next, relevant exogenous signals (step 5), such as electricity price data from markets, must be incorporated for their use in the objective function.

The optimization model is then solved for a specified optimization horizon, which is the duration for which an optimized schedule is to be generated, with a selected temporal resolution (step 6). Based on the temporal resolution, the optimization horizon is segmented into discrete time steps. Subsequently, the optimized schedule is used for the control of the flexible energy resource.

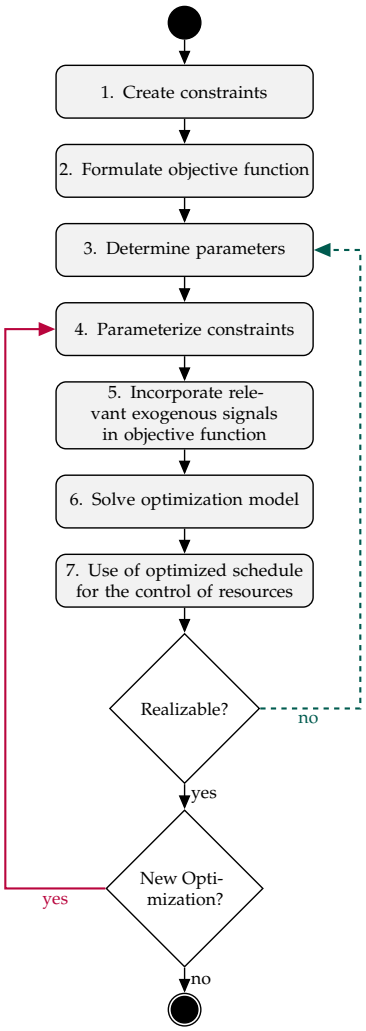



**Figure 1.** Process for the manual generation of optimization models.

Step 7 is not within the scope of this work, as approaches for the manual or automated realization of optimized schedules have already been presented by Wagner et al. [36] Reinpold et al. [37], and Henkel et al. [38], respectively.

Given that time-dependent data such as electricity price data and specific, changeable parameters like targets or initial system states are incorporated into the generated optimization model, it is necessary to regenerate the model for each subsequent use. This ensures that the model remains accurate and relevant under changing conditions. Therefore, to accommodate these updated conditions, the process reinitiates at Step 4 (red arrow), allowing for a systematic update of the model with the latest data and parameters if the optimized schedule was realizable and whenever a new optimized schedule is required ('new optimization?'). This cyclical regeneration is crucial for maintaining the model's effectiveness and reliability in dynamic environments.

If the optimized schedule proves to be unrealizable—for instance, due to changes in operational behavior stemming from degradation—it becomes necessary to determine new parameters. Accuracy indicators that can be utilized for the evaluation of the model performance are presented by Wen et al. [28]. In such instances, restarting the process at step 3 ("no", dashed line) is required to adjust and recalibrate the optimization model with updated parameters.

### 3.2. Model Structure for the Representation of Systems of Flexible Energy Resources

Based on an analysis of existing generic optimization models, a modular model structure has been developed by Wagner et al. [3] to represent *systems* consisting of flexible energy *resources* using different *flexibility features*. Each flexibility feature encapsulates a set of constraints. This allows for the creation of an optimization model for resources within a system depending on the necessary level of detail, as not all flexibility features might be applicable. The validation of the model structure allows for its use without a recurring validation of the model instance. This model structure fulfills Requirement ①.

As shown in Figure 2, the model structure can be used to instantiate an MILP optimization model of a *system* of connected *resources*. Different *resources* within the *system* are connected via so-called *dependencies*, which represent the structure of a system. Dependencies are connections of flows of the output of at least one *resource* with the input of at least one other *resource* [3].

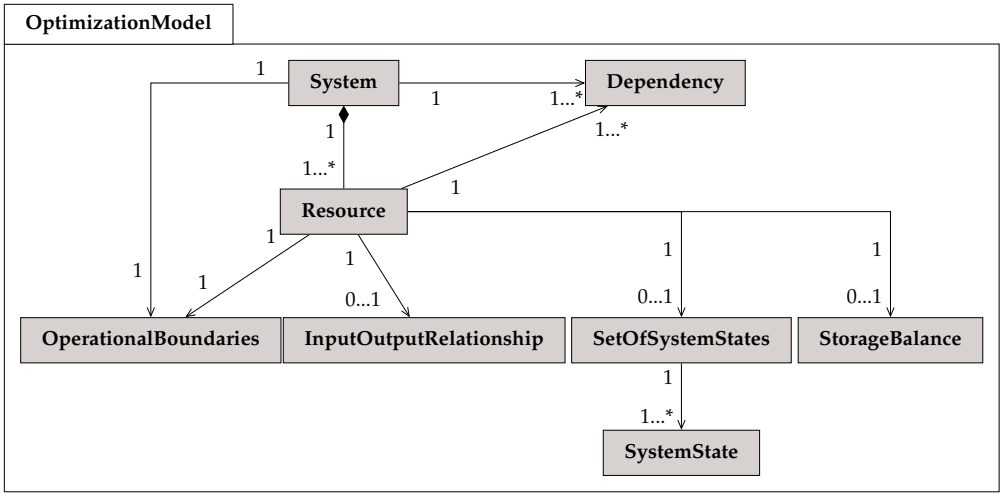



**Figure 2.** Meta-model of the model structure [3], visualized by Wagner and Fay [23]. The numbers represent cardinalities, indicating the minimum and maximum number of relationships between the entities connected by the lines (e.g., 1, 0…1, 1…*, etc., where "*" denotes "many").

There are two types of *dependencies*: either all connected resources of one *dependency* can be connected simultaneously ("correlative dependency"), or only exactly two resources can be connected ("restrictive dependency") at any time step within the optimization horizon, i.e., an XOR connection [3]. An example for the restrictive dependency as a connector of resources is a diverter valve.

The limits of the *system* and of each *resource* are characterized by their respective *operational boundaries* [39–41]. The characteristics of a *resource* can furthermore be described by an *input–output relationship* [39,41] and a *set of system states* [39,40]. Each *system state* is characterized by flow limits [14], holding durations [39], follower states [40], and ramp limits [40]. Furthermore, *Resources* with a storage capacity can also be considered within the optimization model based on a *storage balance* [39,41].

The constraints that are encapsulated within each flexibility feature are listed in Appendix A.1.

### 3.3. Process for the Automatic Generation of Optimization Models

The model generation process outlined in Section 3.1 demonstrates significant potential for automation, particularly because its current reliance on domain-specific knowledge and repetitive user input makes it not universally applicable and prone to errors. Algorithm 1 details the primary function `automaticModelGeneration()` dedicated to the automatic production of an optimization model for a system of flexible energy resources. In this

algorithm, the function `getParameters()` (see Section 3.3.1) is employed to construct a data model *dataModel* that contains all the parameters necessary for an optimization model. These parameters are derived from time series data *tsd* and an information model *fpd* that represents the structure of the system [23].

This data model is subsequently utilized to instantiate and parameterize an optimization model *optModel* within the function `parameterizeOptModel()` (see Section 3.3.2), drawing on the generic model structure (see Section 3.2). Additionally, it is essential to define the optimization horizon $\mathcal{T}$. The process of choosing and designing objective functions falls outside the scope of this work. Hence, specifying an objective function *obj* becomes necessary when it differs from the common objective of minimizing the total cost of electric energy bought from an electricity market [9], as shown in Equation (1), such as the European intra-day market. Individual values for the cost of electric energy per time step $c_{\mathrm{el},t}$ are used within the objective function. For this case, the reference decision variable *var* is typically chosen to be the input flow of the electric energy of the system for energy consumers.

$$obj = \min \sum_{t \in \mathcal{T}} var_t \cdot c_{\mathrm{el},t} \tag{1}$$

**Algorithm 1:** Automatic Generation of an Optimization Model

**Data:** TimeSeriesData *tsd*
**Data:** InformationModel *fpd*
**Data:** ObjectiveFunction *obj*
**Data:** OptimizationHorizon $\mathcal{T}$
**Result:** OptimizationModel *optModel*
**Function** `automaticModelGeneration`(*tsd*, *fpd*, *obj*, $\mathcal{T}$)**:**
    DataModel *dataModel* ← `getParameters`(tsd, fpd) `// see Algorithm 2 in Section 3.3.1`
    optModel ← `parameterizeOptModel`(dataModel, obj, $\mathcal{T}$) `// see Algorithm 3 in Section 3.3.2`
    **return** optModel

### 3.3.1. Derivation of Parameters

Algorithm 2 outlines the function `getParameters()` for automatically deriving parameters. These parameters are essential for instantiating an optimization model, thereby meeting Requirement ②. This methodology has been developed in prior work by Wagner and Fay [23] and is briefly summarized in this section.

The preprocessing of time series data captured from the operation of each resource within the system (via the function `doPreProcessing()`) includes verifying the significance of correlations among different time series data columns and assessing the dataset's stationarity, i.e., assessing whether the statistical properties of the time series data remain constant over time. If the statistical analysis determines that the time series dataset is insufficient for deriving parameters, e.g., because the relationships are not significant or the time series data are non-stationary, a new dataset must be selected, and no further functions are executed. In this case, Algorithm 2 returns `null`, and a new dataset needs to be imported and newly assessed. If the dataset qualifies for parameter derivation, any inaccurate values, such as outliers, are statistically identified and corrected to guarantee a high-quality dataset [23].

Subsequently, parameters are derived that describe each resource's behavior, incorporating them into the data model *dataModel* through the function `addResource()`. This parameter set and the accompanying data model was designed to fit the generic model

structure described in Section 3.2 and encompass operational boundaries, parameters detailing the input–output relationship, and parameters related to system states, such as flow limits, follower states, holding durations, and ramp limits. Additionally, if a resource has a storage capacity, parameters describing this aspect are also incorporated through conversion efficiencies for (dis-)charging. The data model also contains information regarding dependencies within the system via two lists for resources involved in one dependency and an attribute for the dependency's type. Furthermore, system-specific parameters, i.e., the operational boundaries of the system as a whole, and model-specific parameters, such as the optimal temporal resolution for balancing computational complexity with an accurate depiction of resource characteristics, are also part of the data model [23].

**Algorithm 2:** Derivation of Parameters [23]

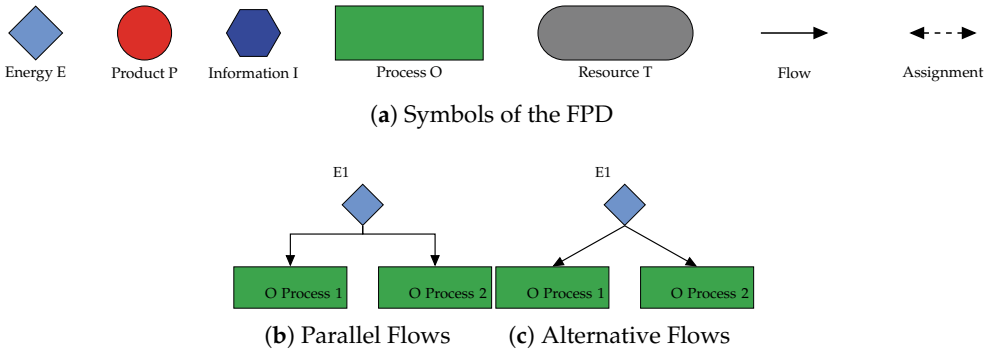
```
Data: TimeSeriesData tsd
Data: InformationModel fpd
Result: DataModel dataModel
Function getParameters(tsd, fpd):
    for resourceTsd : tsd do
        resourceTsdPrePro ← doPreProcessing(resourceTsd)
        if resourceTsdPrePro == null then
            return null
        dataModel.addResource(getResourceParameters(resourceTsdPrePro))
    dataModel.addSystemParameters( getSystemParameters(
     doPreProcessing(tsd.system), dataModel))
    dataModel.addDependencies(extractDependencies(fpd))
    return dataModel
```

The function `getResourceParameters()` employs various algorithms to derive the specified parameters: it identifies operational boundaries by calculating the minimum and maximum values within a column of time series data. Parameters that describe the input–output relationship are ascertained through (piecewise) linear regression, evaluating whether piecewise linear approximation (PLA) is necessary using a threshold value for the coefficient of determination ($R^2$) to keep the computational complexity of the optimization model as low as possible. Additionally, system states are identified from time series data by utilizing a hidden Markov model. Then, system state-related parameters are calculated based on the identified system states: holding durations are determined by extracting the minimum and maximum durations of state activity. The follower states for each state are identified by analyzing the sequence of active states. Ramp limits per state are established by examining the rate of change of the input flow over time. The minimum and maximum storage capacity is determined analogously to the operational boundaries. The efficiencies of the storage charging and discharging are calculated based on the average quotient of the flows at the input and output port, respectively, and inside the storage tank [23].

Additionally, the operational boundaries of the system and the optimal temporal resolution are determined via the function `getSystemParameters()` and integrated into the data model. Furthermore, resolution-dependent parameters, namely the minimum and maximum holding durations for all resources and each of their system states, are recalculated from time units to the corresponding number of time steps. A temporal resolution is chosen to convert the time duration to time steps without introducing inaccuracies through the conversion of holding durations from time in hours to time steps or overly increasing computational complexity. Therefore, the data model is also an input parameter for this function [23].

The system's structure is encapsulated within an formalized process description (FPD)-based information model *fpd*, which undergoes analysis to extract resource dependencies through the function `extractDependencies()` [23].

The modeling concept of the FPD [42] consists of states, namely energy, product, and information, as well as processes and resources, as shown in Figure 3a. The corresponding symbols are used to visually model the structure of the system by connecting states and processes via flows. Two types of flows can be used within the FPD modeling concept: parallel (Figure 3b) and alternative flows (Figure 3c). Resources are then assigned to processes [42].




(**a**) Symbols of the FPD

(**b**) Parallel Flows (**c**) Alternative Flows

**Figure 3.** Modeling concept of the FPD [42], including symbols and flow types.

During the extraction of dependencies using the function `extractDependencies()`, connected processes are identified, and connections between resources within the system are determined via resources assigned to processes. The type of dependency is categorized as either 'correlative' or 'restrictive'. The correlative dependency indicates that all linked resources can be concurrently connected, modeled via parallel flows (see Figure 3b). The restrictive dependency is represented by an XOR connection, indicating that only one flow can actively link exactly two resources at any given time. This is modeled through the alternative flow (see Figure 3c). The extracted dependencies are then also incorporated into the data model (function `addDependencies()`) [23].

Algorithm 2 returns the data model, which is subsequently used by Algorithm 3.

#### 3.3.2. Automatic Generation of a Parameterized Optimization Model Instance

This section describes the function `parameterizeOptModel()` that is detailed in Algorithm 3 to automatically create a parameterized instance of the optimization model structure (presented in Section 3.2) for the fulfillment of Requirement ③. As shown in Algorithm 1, the data model created through the application of Algorithm 2 is the main input of Algorithm 3.

Initially, the optimization horizon (`setHorizon()`) and the temporal resolution (`setTemporalResolution()`) are set. The objective function (`addObjectiveFunction()`) is also added.

As described in Section 3.2, a system consists of one to many resources. Necessary decision variables for each flexibility feature are created via the function `addDecVars(name of resource, type of flexibility feature)` for their representation of system or resource characteristics. This function adds an array of decision variables for a specific variable type/flexibility feature, such as 'input flow' (Algorithm 3, line 8) or 'system states' (Algorithm 3, line 15), to the optimization model. Decision variables are created for both the system as a whole (input and output flows, lines 5 and 6) as well as for each of the resources individually. Decision variables represent the input and output flows of the system and of individual resources for the optimization and may also include a variable to

represent the storage capacity of a resource, modeled as the SOC, if the respective resource exhibits a storage capacity. Each element of a decision variable array represents one time step within the optimization horizon, except for decision variables for system states and storage capacities. In these cases, the number of decision variables is (number of time steps + 1) to accommodate an initial state or an initial SOC.

**Algorithm 3:** Automatic Generation of an Optimization Model Instance

```
   Data: DataModel dataModel
   Data: ObjectiveFunction obj
   Data: OptimizationHorizon 𝒯
   Result: OptimizationModel optModel
1  Function parameterizeOptModel(dataModel, obj, 𝒯):
2      optModel.setHorizon(𝒯)
3      optModel.setTemporalResolution(dataModel.temporalResolution)
4      optModel.addObjectiveFunction(obj)
5      optModel.addDecVars(SYSTEM, input)
6      optModel.addDecVars(SYSTEM, output)
7      for resource : dataModel.resources do
8          optModel.addDecVars(resource.name, input)
9          optModel.addDecVars(resource.name, output)
10         if resource.io != null then
11             if resource.io.pla==true then
12                 optModel.addDecVars(resource.name, io)
13             optModel.addFlexibilityFeature(resource.name,
                input-output-relationship)
14         if resource.systemStates != null then
15             optModel.addDecVars(resource.name, systemStates)
16             optModel.addFlexibilityFeature(resource.name, systemStates)
17         if resource.storage != null then
18             optModel.addDecVars(resource.name, storage)
19             optModel.addFlexibilityFeature(resource.name, storage)
20     for dep : dataModel.dependencies do
21         if dep.type==RESTRICTIVE then
22             optModel.addDecVars(dependency, restrictiveDependency{dep})
23         optModel.addFlexibilityFeature(dependency, dep.type{dep})
24     return optModel
```

The flexibility feature *operational boundaries* is already applied during the creation of decision variables for input and output flows of the system (Algorithm 3, lines 5 and 6) and resources (Algorithm 3, lines 8 and 9), as a value range is set that corresponds to the lower and upper boundaries. Similarly, the minimum and maximum capacities of storage-type resources are set as value ranges for SOC decision variables (Algorithm 3, line 18), if applicable. The value ranges are set using the respective parameters contained within the data model.

As described in Algorithm 3, the applicable flexibility features, identified based on the existence of the respective parameters within the data model, e.g., in line 10, are instantiated and parameterized by a function `addFlexibilityFeature(name of resource, type of flexibility feature)` using parameters contained within the data model created using

Algorithm 2. This function selects and parameterizes the constraints contained within the flexibility feature and adds them to the optimization model *optModel* for each resource included in the data model.

The adoption of certain flexibility features requires the introduction of additional decision variables: Additional decision variables comprise binary variables for the modeling of a *set of system states* (Algorithm 3, line 15) and for segments of piecewise linear approximation (PLA, Algorithm 3, line 12) within the flexibility feature *input–output relationship*. The PLA within the flexibility feature *input–output relationship* also necessitates an array of continuous decision variables (see Equations (A2)–(A5) [3].)

Dependencies among resources within the system, saved to the data model through Algorithm 2, are also added to the optimization model. 'Restrictive' type dependencies require the generation of additional decision variables for resources involved in one dependency *dep* (Algorithm 3, line 22). These are essential to accurately model the exclusivity of a flow of dependencies of the 'restrictive' type between resources within the optimization model. For each restrictive dependency, the creation of two types of decision variables is required: one array of binary variables per resource to signify the activation of the port (input/output), and one array of continuous variables to represent the connection of flows between two resources [3].

The function `parameterizeOptModel()` described in Algorithm 3 returns the optimization model, which can then be solved to generate an optimized schedule. This function needs to be executed for each new optimization horizon (see Section 3.1).

Changes in operational characteristics over time, such as the degradation of energy storage systems or electrolyzers, are not explicitly accounted for in the model generation process. However, these changes can cause deviations between model predictions, i.e., the optimized schedule, and actual resource behavior, potentially leading to infeasible schedules. To this end, as outlined in Section 3.1, the function described in Algorithm 1 can be employed to recreate a model instance either cyclically or on demand, ensuring the model remains aligned with current resource conditions and operational capabilities.

Requirement ④ indicates that it is necessary to guarantee the traceability and comprehensibility of the generated model. Traceability and comprehensibility, in this context, mean that the model is not a black box model, i.e., a model where structures can be recognized and parameters have physical meanings. The sequential model generation process, summarized in Algorithm 1, allows users to easily verify each stage of the process. The data model, generated through the application of Algorithm 2, contains all parameters used for the parameterization of the optimization models. Users can inspect this set of parameters to fulfill the stated goals. This can include a step to manually verify or modify certain or all aspects.

Specifically, the inclusion of operational boundaries, input–output relationships, and the consideration of system states is essential. Operational boundaries and input–output relationships, commonly known by resource operators, enhance the comprehensibility through recognition. System states, reflected in resource control mechanisms, also play a crucial role in understanding the model's behavior and ensuring its efficient performance. These aspects directly contribute to the traceability and comprehensibility of the model and should be incorporated into the evaluation.

Thanks to the modular generic optimization model structure (see Section 3.2), the use and effect of individual model components (flexibility features) can also be verified by the end-users. This has already been demonstrated by Wagner et al. [3]. Therefore, Requirement ④ is implicitly fulfilled through the methodology outlined in Section 3.

### *3.4. Implementation of the Algorithms*

Algorithm 1, as well as the functions described in Algorithms 2 and 3, have been implemented in Java (The implementation is available at https://github.com/lukas-wagner/AutoModelGeneration, accessed on 6 January 2025). Parts of Algorithm 2—in particular, the PLA and the clustering algorithm—have been implemented by interfacing R and Python, respectively. The optimization model structure [3] has also been implemented in Java for the solution with IBM ILOG CPLEX, utilizing CPLEX Concert Technology [43].

The FPD-based information models for the representation of the structure of the systems have been built and serialized using the web-based modeling tool (https://demo.fpbjs.net, accessed on 6 January 2025) developed by Nabizada et al. [44]. The `.json` file serves as an input and is analyzed using the function `extractDependencies()` contained in Algorithm 2.

## 4. Evaluation of the Methodology

This section demonstrates the applicability of the methodology presented in Section 3 to the automatic generation of optimization models for systems of flexible energy resources.

As described in Section 3.3, the process for the automatic generation comprises two distinct phases—parameter derivation (Requirement ②) and parameterization of an optimization model instance (Requirement ③)—and is constructed in a traceable way (Requirement ④). The efficacy and validity of the parts of the methodology corresponding to Requirement ① (model structure) and Requirement ② (parameter derivation) have already been established through earlier research [3,23,36–38]:

The integrity of the generic optimization model structure and its suitability for depicting systems of flexible energy resources have been validated by Wagner et al. [3] and further confirmed in a previous study by the authors [23,36–38]. Specifically, this model structure was applied to optimizing systems of electrolyzers [3,38], an experimental distillation unit [37], a combined heat and power (CHP) system [23], and a waste water treatment plant [36], demonstrating its effectiveness and applicability to various types of energy resource systems. During the evaluation of the model structure, the correct instantiation of constraints encapsulated within one flexibility feature was also demonstrated [3].

The methodology for parameter derivation (Requirement ②), as outlined in Algorithm 2, was developed and validated by Wagner and Fay [23]. The parameter derivation methodology was also applied for the derivation of parameters in the work of Henkel et al. [38] for a system of electrolyzers.

Therefore, this section presents the validation of the process for the automatic generation of a parameterized instance of the generic optimization model, outlined in Algorithms 1 and 3 (corresponding to Requirement ③). The validity of Algorithms 1 and 3 is demonstrated by two case studies of a flexible electric power generation system and a flexible consumption system. The setup of the validation is outlined in Section 4.1. The case studies are presented in Sections 4.2 and 4.3. This section concludes with a summary of the results in Section 4.4. Section 4.4 also describes the fulfillment of Requirement ④, i.e., the traceability and comprehensibility of the generated optimization models.

### *4.1. Setup of the Evaluation*

As outlined in the previous section, certain aspects of the methodology have already been validated. Consequently, the validation in this work focuses on evaluating the accuracy of the generated decision variables, the appropriate selection of flexibility features (sets of constraints), their correct parameterization, and ultimately, the generation of a feasible optimization model.

The results from deriving parameters are presented separately from the creation of a model instance to improve the clarity of each section, aligning with the functions described in Algorithms 2 and 3. However, as illustrated in Algorithm 1, these two functions are actually integrated within the main function `automaticModelGeneration()` to facilitate an automatic model generation process without manual intervention.

The following sections are structured similarly:

First, the accuracy of the parameter set is evaluated based on the normalized root mean square error (nRMSE) of a validation time series data set and the parameters used in an optimization model with predetermined system input [23].

Following, the correctness of the model instance is evaluated by comparing a manually created reference model (subsequently referred to as 'expected model') to the automatically generated model instance. The definition of the expected model is achieved by analyzing the parameter set and manually selecting the necessary flexibility features and associating parameters. For this purpose, an optimization horizon of 10 hours is selected.

The feasibility of the automatically generated optimization model is demonstrated by solving each model for the previously selected optimization horizon, with the temporal resolution determined by the function `getSystemParameters()` in Algorithm 2. The objective function incorporates the dynamic electricity prices illustrated in Figure 4. The solver's default optimality gap of $10^{-4}$ is used. The optimality gap is a measure of how close a found solution, i.e., the optimized schedule, is to the theoretically optimal solution.

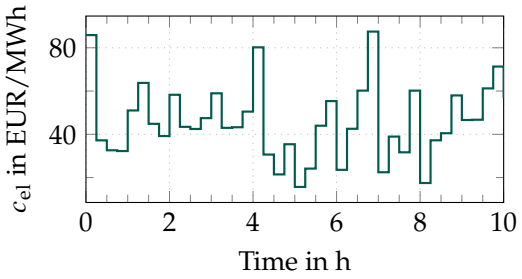



**Figure 4.** Electricity price of the continuous intra-day market [45].

All case study-related calculations are carried out on Windows 11 with an AMD Ryzen 7 PRO 5850U processor and 32 GB of RAM. IBM ILOG CPLEX is used as a solver (see Section 3.4).

### *4.2. Automatic Generation of Optimization Model of CHP System*

This section describes the application of the method for the automatic generation of an optimization model presented in Section 3 to a CHP system depicted in Figure 5. The system comprises two resources: a gas-fired generator and a heat exchanger. First, the derived parameters are described (Section 4.2.1), then the generated model instance is verified (Section 4.2.2). Finally, the feasibility of the optimization model instance is demonstrated in Section 4.2.3.

#### 4.2.1. Derived Parameters of CHP System

This section describes the results of the application of the algorithm for parameter derivation (Algorithm 2) for the CHP system depicted in Figure 5. The results reported in this subsection have been previously established as part of the validation in the work of Wagner and Fay [23].

The time series data used in the following have been captured from resource operation. During preprocessing, erroneous values have been replaced while retaining the overall integrity of the dataset, including its size of 30,000 datapoints (1 month, with a temporal resolution of 1 min) [23].

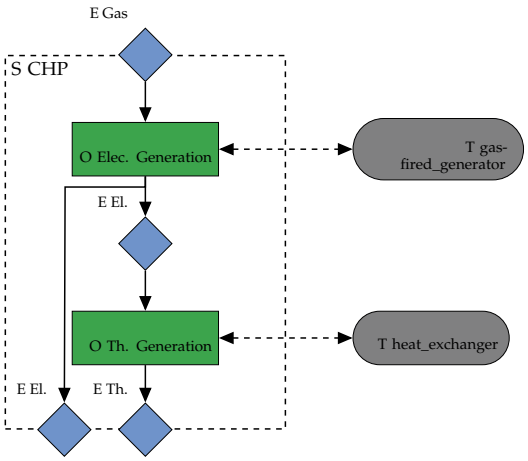



**Figure 5.** FPD information model of the CHP system [23].

Tables detailing the content of the data model, specifically the parameters derived using Algorithm 2 from the preprocessed time series data, are provided below. These include operational boundaries and parameters, i.e., slope and intercept, that describe the linearized input–output relationship of each resource (see Table 2).

**Table 2.** Parameters of resources within the CHP system [23].

| Subset | Parameter | Gas-Fired Gen. | Heat Ex. |
|---|---|---|---|
| Operational boundaries (input) | Lower bound, kW | 0 | 0 |
| | Upper bound, kW | 2912 | 1101 |
| Operational boundaries (output) | Lower bound, kW | 0 | 0 |
| | Upper bound, kW | 1101 | 1746 |
| IO relationship | Slope, kW/kW | 0.38 | 1.15 |
| | Intercept, kW | −18.4 | 55.2 |

Additionally, two discrete system states $s$ for the generator were identified using a clustering algorithm, and system state-related parameters (boundaries, minimum and maximum holding duration, follower states, and ramp limits) are listed in Table 3. The heat exchanger does not have any system states. These parameters were derived using the function `getResourceParameters` of Algorithm 2 [23].

**Table 3.** System states and related parameters of the gas-fired generator within the CHP system [23].

| State $s$ | 0 | 1 |
|---|---|---|
| $P_{\text{in., min},s}$, kW | 0 | 0 |
| $P_{\text{in, max},s}$, kW | 2912 | 1452 |
| $P_{\text{out, max},s}$, kW | 1101 | 536 |
| $t_{h,\text{min},s}$, h | 8.15 | 0.5 |
| $t_{h,\text{max},s}$, h | ∞ | ∞ |
| $S_{F,s}$ | {1} | {0} |
| $\text{ramp}_{\text{input, min},s}$, kW/h | 0 | 0 |
| $\text{ramp}_{\text{input, max},s}$, kW/h | 128,587 | ∞ |

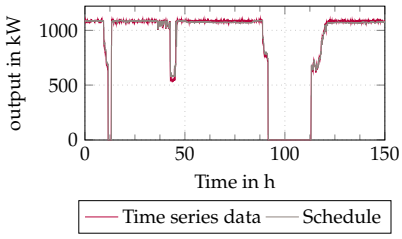
Dependencies were extracted from the FPD-based information model of the CHP system, as depicted in Figure 5, using the function `extractDependencies()` included in Algorithm 2 (see Table 4) [23].

**Table 4.** Dependencies of resources within the CHP system (Figure 5) [23].

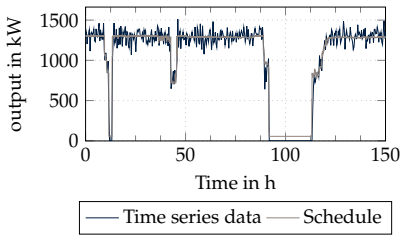
| Dependency | Type | Output Resources | Input Resources |
|---|---|---|---|
| 1 | Correlative | System input | Gas-fired gen. |
| 2 | Correlative | Gas-fired gen. | System output |
| 3 | Correlative | Gas-fired gen. | Heat ex. |
| 4 | Correlative | Heat ex. | System output |

To optimally balance computational complexity and accurately match holding durations, a temporal resolution of 0.125 h was automatically selected [23].

A simulation case study compares these parameters to time series data and demonstrates very good alignment (see Figure 6), indicated by low nRMSE values (1% for the gas-fired generator, 6% for the heat exchanger) [23].




(**a**) Output of gas-fired generator




(**b**) Output of heat exchanger

**Figure 6.** Comparison of generated schedules with corresponding time series data of resources within the CHP system [23].

#### 4.2.2. Verification of the Generated Model Instance of the CHP System

This section describes the application of Algorithm 3 for the CHP system and verifies the generated model. The data model presented in the previous section acts as the main input for the generation of an optimization model instance (`parameterizeOptModel()`), as shown in Algorithm 1.

The generation of the model instance is realized through the functions `addFlexibilityFeature()` and `addDecVars()`, respectively. The resulting optimization model instance (this model instance is available at https://github.com/lukas-wagner/AutoModelGeneration, accessed on 6 January 2025) is analyzed below.

As described in Section 3.2, continuous (cont.) decision variables are to be created for system inputs and outputs, resource inputs and outputs, and binary decision variables for system states. The length of one decision variable array is determined by the duration of the optimization horizon (10 h) and the temporal resolution (0.125 h), thus resulting in

80 time steps. The decision variables for system states include one additional time step to account for starting states; thus, these variables must be of length 81.

Table 5 shows a comparison of the expected and generated decision variables. Therein, it is shown that all expected decision variables have been created at the expected length, and the right type was chosen: continuous decision variables are created for input and output flows, for both resources as well as for the system. Additionally, binary decision variables are created for the two system states of the generator.

**Table 5.** Comparison of expected and generated decision variables of the CHP system.

| Resource | Expected Variable | Type | Length | Generated |
|---|---|---|---|---|
| System | Input | Cont. | 80 | yes |
| System | Output (el.) | Cont. | 80 | yes |
| System | Output (th.) | Cont. | 80 | yes |
| Generator | Input (el.) | Cont. | 80 | yes |
| Generator | Output (th.) | Cont. | 80 | yes |
| Generator | System State 0 | Binary | 81 | yes |
| Generator | System State 1 | Binary | 81 | yes |
| Heat exchanger | Input (th.) | Cont. | 80 | yes |
| Heat exchanger | Output (th.) | Cont. | 80 | yes |

Analogously, Table 6 shows a comparison of the expected and instantiated flexibility features. The instantiation of constraints encapsulated within flexibility features and their correct parameterization is verified for each applicable flexibility feature. As shown in the parameter set presented in Section 4.2.1, flexibility features for *operational boundaries* (Equation (A1)), a *linearized input–output relationship* (Equation (A2) with $|\mathcal{K}| = 1$) as well as *system states* (state selection) and associated features (holding durations, follower states, ramp limits) are necessary (Equation (A7)ff). For the heat exchanger, only flexibility features *operational boundaries* and a *linearized input–output relationship* are necessary. Flexibility features related to system states are not applicable for the heat exchanger, as no system state-related parameters are contained. As extracted from the FPD information model shown in Figure 5, the system exhibits four *correlative dependencies* (see Table 4 and Equation (A15)).

**Table 6.** Comparison of expected and instantiated flexibility features for the CHP system.

| Resource | Expected Feature | Inst.? | Para.? |
|---|---|---|---|
| System | Operational boundaries (input) | yes | yes |
| System | Operational boundaries (output el.) | yes | yes |
| System | Operational boundaries (output th.) | yes | yes |
| Generator | Operational boundaries (input) | yes | yes |
| Generator | Operational boundaries (output) | yes | yes |
| Generator | Linear input–output relationship | yes | yes |
| Generator | System state selection | yes | yes |
| Generator | State sequences | yes | yes |
| Generator | Holding durations | yes | yes |
| Generator | Ramp limits | yes | yes |
| Heat exchanger | Operational boundaries (input) | yes | yes |
| Heat exchanger | Operational boundaries (output) | yes | yes |
| Heat exchanger | Linear input–output relationship | yes | yes |
| System | Correlative dependency: input–generator | yes | yes |
| System | Correlative dependency: generator–heat ex. | yes | yes |
| System | Correlative dependency: generator–output el. | yes | yes |
| System | Correlative dependency: heat ex.–output th. | yes | yes |

Inst.—Instantiated. Para.—Parameterized.

As shown in Table 6, all expected elements have been correctly instantiated and parameterized using the parameter set previously derived (see Section 4.2.1).

The analyses in Tables 5 and 6 show that an accurate optimization model instance of the CHP system has been generated.

#### 4.2.3. Demonstration of Feasibility of Model Instance of CHP System

In addition to the model being used for the verification of the accuracy of the parameter set, the feasibility of the optimization model is further validated by solving it to generate an optimized schedule. A target of 85% capacity utilization is established, which corresponds to the generation of 6000 kWh of electric energy during the optimization horizon. As described in Section 3.1, this ensures the generation of a useful schedule. The constraint to enforce a target is also part of the optimization model structure (Equation (A6)) and can be set by adding the respective parameters (target energy amount, target decision variable) to the data model.

Figure 7 displays the optimized schedule generated by solving the optimization model instance, which aims to maximize the total profit from the sale of electric energy $c_{\text{el}}$ (see Figure 4; output of electric energy is output of the generator in time interval $P \cdot \Delta t$) to the continuous intra-day market. This requires the definition of an objective function (see Equation (2)). The formulation of this objective function is based on the assumption that the CHP system acts as a price taker and that the cost of natural gas input remains constant and can therefore be omitted in this objective function.

$$obj = \max \sum_{t \in \mathcal{T}} P_{\text{generator, out},t} \cdot c_{\text{el},t} \tag{2}$$

The optimized schedule displayed in Figure 7 demonstrates the feasibility of the automatically generated optimization model. The energy-flexible operation of the CHP system yields a monetary benefit of EUR 344.47 from the sale of electric energy. The cost of natural gas must be deducted from this income. Constant operation with an identical target of 6000 kWh of electric energy (=600 kW output of electric power) of the CHP system, with identical price values, as shown in Figure 4, within the operating period under consideration would have yielded a monetary benefit of EUR 65.5. The computational time was 0.038 s, mostly due to the low complexity of the model and short optimization horizon.

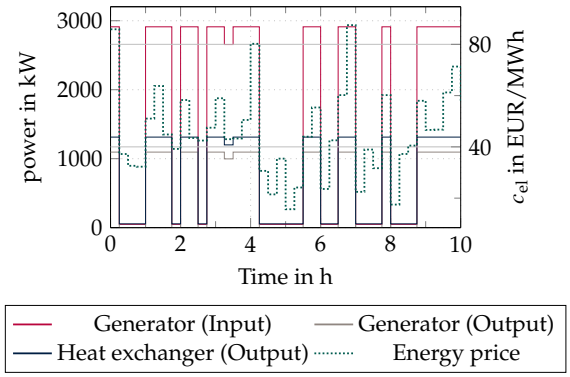



**Figure 7.** Optimized schedule for CHP system.

The input of the heat exchanger is not displayed in Figure 7, as it is equal to the output of the generator, as shown in Figure 5. The generated operational schedule has been subjected to a manual, offline analysis. All constraints that are part of the instantiated flexibility features, as detailed in Table 6, have been considered correctly during the generation of the optimized schedule. This includes input–output relationships, system

state selection, follower states, holding durations, and ramp limits, as well as dependencies within the system.

### *4.3. Automatic Generation of Optimization Model of Refrigeration System*

This section describes the second case study, applying the methodology outlined in Section 3 to a refrigeration system. The structure of the following subsections is similar to the previous sections: first, the results of the derivation of parameters are described (Section 4.3.1), then the generated optimization model instance is validated (Section 4.3.2). The feasibility of the optimization model instance is demonstrated in Section 4.3.3.

The dataset used for this case study has been made available by Cirera et al. [46], which was also used in the work of Cirera et al. [47].

As shown in Figure 8, the system consists of two vapor-compression refrigeration machines that work in tandem: Each compressor is responsible for compressing the refrigerant vapor, increasing its pressure and temperature. The high-pressure vapor then moves to a condenser, where it releases heat to the environment and condenses into a high-pressure liquid. This liquid refrigerant is then expanded and cooled by the thermal expansion valves before entering the evaporators, where it absorbs heat and evaporates back into vapor. The cycle continuously repeats. For generating a useful optimization model of the refrigeration machines as flexible energy resources, some intermediate steps of the vapor-compression cycle do not need to be modeled explicitly. It is sufficient to model the refrigeration machines as resources that convert electrical energy (E El.) directly into cooling energy (E. Th.).

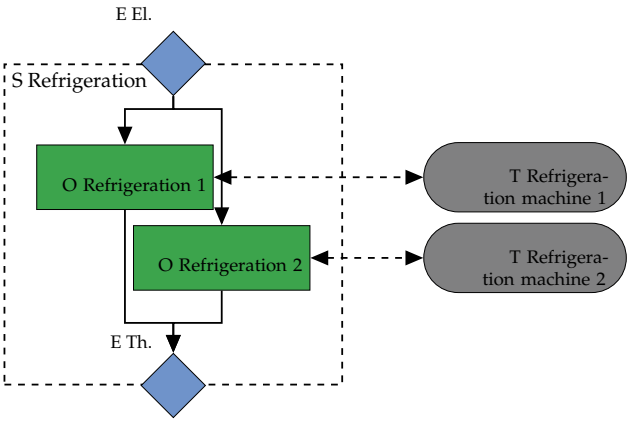



**Figure 8.** FPD information model of the refrigeration system.

#### 4.3.1. Derived Parameters of Refrigeration System

This section presents the parameters derived via the application of Algorithm 2 for the refrigeration system.

The length of the time series dataset was 30,000 data points (roughly 1 month at 1 min time steps), which was deemed to be sufficiently long to capture a variety of operational characteristics. A threshold value of 0.9 for the $R^2$ was set to determine whether a linearized input–output relationship is sufficient [23]. For $R^2$ values below this threshold, PLA would be necessary [23]. However, for both refrigeration machines, the $R^2$ was above the threshold (refrigeration machine 1: 0.904, refrigeration machine 2: 0.957). Thus, linearized input–output relationships are sufficient for both refrigeration machines, and parameters for slope and intercept were derived (see Table 7).

Three system states were identified for each refrigeration machine, and associated parameters for limits, holding durations, follower states, and ramp limits were derived (see Table 8).

For the system, operational boundaries for power flows (see Table 7) and dependencies were extracted from the FPD information model shown in Figure 8. As shown in Table 9, two dependencies were extracted.

**Table 7.** Parameters of the refrigeration system.

| **Subset** | **Parameter** | **System** |
|---|---|---|
| Operational boundaries (input) | Lower bound, kW | 0 |
| | Upper bound, kW | 758.86 |
| Operational boundaries (output) | Lower bound, kW | 0 |
| | Upper bound, kW | 2415.82 |
| **Subset** | **Parameter** | **RM1** | **RM2** |
| Operational boundaries (input) | Lower bound, kW | 0 | 0 |
| | Upper bound, kW | 460.98 | 408.57 |
| Operational boundaries (output) | Lower bound, kW | 0 | 0 |
| | Upper bound, kW | 1598.13 | 1367.17 |
| IO relationship | Slope, kW/kW | 3.95 | 2.46 |
| | Intercept, kW | −185.81 | 0.93 |

RM—refrigeration machine.

**Table 8.** System states and related parameters of resources within refrigeration system.

| **Refrigeration Machine 1** | | | |
|---|---|---|---|
| **State** $s$ | **0** | **1** | **2** |
| $P_{\text{in., min},s}$, kW | 0 | 200.51 | 274.21 |
| $P_{\text{in, max},s}$, kW | 199.88 | 273.59 | 460.98 |
| $P_{\text{out, max},s}$, kW | 457.86 | 1212.68 | 1598.13 |
| $t_{h,\text{min},s}$, timesteps | 0 | 0 | 0 |
| $t_{h,\text{max},s}$, timesteps | ∞ | ∞ | ∞ |
| $\mathcal{S}_{F,s}$ | {1, 2} | {0, 2} | {0, 1} |
| $\text{ramp}_{\text{input, min},s}$, kW/h | 0 | 0 | 12.4 |
| $\text{ramp}_{\text{input, max},s}$, kW/h | 794.52 | 1442.99 | 6370.89 |
| **Refrigeration Machine 2** | | | |
| **State** $s$ | **0** | **1** | **2** |
| $P_{\text{in., min},s}$, kW | 0 | 202.69 | 274.07 |
| $P_{\text{in, max},s}$, kW | 199.5 | 273.55 | 408.57 |
| $P_{\text{out, max},s}$, kW | 0 | 944.91 | 1367.17 |
| $t_{h,\text{min},s}$, timesteps | 0 | 0 | 0 |
| $t_{h,\text{max},s}$, timesteps | ∞ | 6 | ∞ |
| $\mathcal{S}_{F,s}$ | {1, 2} | {0, 2} | {0, 1} |
| $\text{ramp}_{\text{input, min},s}$, kW/h | 0 | 18.9 | 8.54 |
| $\text{ramp}_{\text{input, max},s}$, kW/h | 2699.03 | 1765.04 | 6810.14 |

**Table 9.** Dependencies of resources within the refrigeration system (Figure 8).

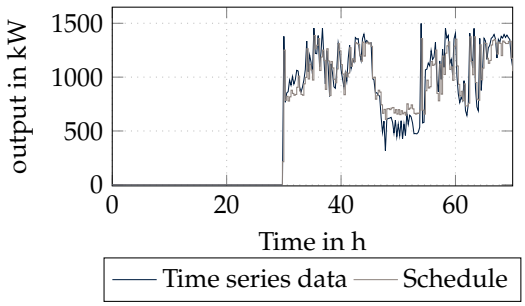
| Dependency | Type | Output Resources | Input Resources |
|---|---|---|---|
| 1 | Correlative | System input | RM1, RM2 |
| 2 | Correlative | RM1, RM2 | System output |

Furthermore, a temporal resolution of 0.25 h was selected. This was done to optimally balance computational complexity and accurately match holding durations of all system states [23].

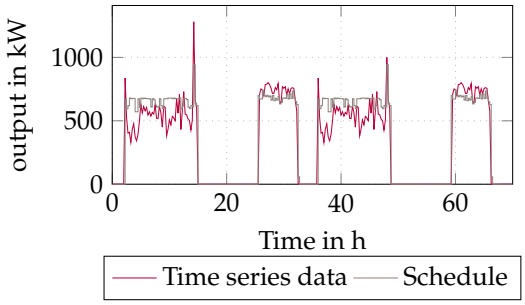
A full interpretation of the parameters is omitted, as the methodology for parameter derivation has already been validated and is applied in this case study for the purpose of generating an optimization model in the following section.

Similar to the validation of the accuracy of the parameters in the work of Wagner and Fay [23], this section validates the parameter set by solving the parameterized optimization model in the style of a simulation model, eliminating the need for an objective function. For comparability, the input time series data for each refrigeration machine are directly linked to the corresponding decision variables. For this validation, previously unused time series data are used (validation dataset). The choice of system states and associated constraints, such as the holding durations for each state, significantly influences the resultant schedule, as these constraints must be adhered to. As outlined in Section 3.2, these constraints encompass the selection of system states, adherence to holding durations, and observance of ramp limits. Furthermore, the maintenance of the input–output relationship, compliance with operational boundaries, and dependencies of resources are also demonstrated.

The resulting comparison is shown in Figure 9. This figure shows the output of each of the refrigeration machines. Therein, a very good alignment can be seen, measured with nRMSE values of 5.22% (refrigeration machine 1, Figure 9a) and 5.34% (refrigeration machine 2, Figure 9b). This not only validates the accuracy of the parameter set but also further validates the methodology for parameter derivation (Algorithm 2).




(**a**) Output of refrigeration machine 1




(**b**) Output of refrigeration machine 2

**Figure 9.** Comparison of generated schedules with corresponding time series data of refrigeration machines within refrigeration system.

#### 4.3.2. Verification of the Generated Model Instance of the Refrigeration System

In this section, the same verification approach is used as described in Section 4.2.2, and an expected model is compared to the generated model (this model instance is available at https://github.com/lukas-wagner/AutoModelGeneration, accessed on 6 January 2025).

Table 10 shows a comparison of the expected and generated decision variables. Continuous decision variables are necessary for system and resource inputs and outputs.

**Table 10.** Comparison of expected and generated decision variables of the refrigeration system.

| Resource | Expected Variable | Type | Length | Generated |
|---|---|---|---|---|
| System | Input | Cont. | 40 | yes |
| System | Output (th.) | Cont. | 40 | yes |
| RM 1 | Input (el.) | Cont. | 40 | yes |
| RM 1 | Output (th.) | Cont. | 40 | yes |
| RM 1 | System State 0 | Binary | 41 | yes |
| RM 1 | System State 1 | Binary | 41 | yes |
| RM 1 | System State 2 | Binary | 41 | yes |
| RM 2 | Input (el.) | Cont. | 40 | yes |
| RM 2 | Output (th.) | Cont. | 40 | yes |
| RM 2 | System state 0 | Binary | 41 | yes |
| RM 2 | System state 1 | Binary | 41 | yes |
| RM 2 | System state 2 | Binary | 41 | yes |

RM—refrigeration machine.

For both refrigeration machines, binary decision variables are also needed for the representation of each of the system states. In Table 10, it is shown that all expected decision variables have been created at the expected length, and the right type was chosen. This was achieved, as described in Algorithm 3, through an analysis of the data model (for contents, see Tables 7–9).

Table 11 shows the comparison of expected and instantiated flexibility features to verify the correct instantiation of flexibility features and their correct parameterization. The expected flexibility features for both refrigeration machines are identical, as a similar set of parameters was derived (see Tables 7–9). The expected model contains *operational boundaries* for the system as well as for both refrigeration machines (Equation (A1)). Both refrigeration machines are represented by a *linearized input–output relationship* (Equation (A2) with $|\mathcal{K}| = 1$) as well as *system states* and associated constraints (Equation (A7)ff). Furthermore, two *dependencies* for the representation of the system structure were extracted from the FPD information model (see Figure 8 and Equation (A15)).

As shown in Table 11, all expected flexibility features were instantiated as well as parameterized.

The analyses in Tables 10 and 11 demonstrate the accuracy of the automatically generated optimization model instance for the refrigeration system.

#### 4.3.3. Demonstration of Feasibility of Model Instance of Refrigeration System

In this section, an optimized schedule is calculated for the refrigeration system using the optimization model that was automatically generated and subsequently verified in Section 4.3.2. This is done to demonstrate the feasibility of the optimization model. The optimization model was solved using the default objective function to minimize the total cost of electric energy, shown in Equation (1), with price data shown in Figure 4. With a target of 6000 kWh of thermal energy for each of the refrigeration machines (set as described in Section 4.2.3), the total cost of electric energy was EUR 157.1. A constant operation of the refrigeration system with the same target within the operating period under consideration

would have resulted in an electric energy cost of EUR 202.23. The computational time was 0.2 s.

**Table 11.** Comparison of expected and instantiated flexibility features for refrigeration system.

| Resource | Expected Feature | Inst.? | Para.? |
|---|---|---|---|
| System | Operational boundaries (input) | yes | yes |
| System | Operational boundaries (output) | yes | yes |
| RM 1 | Operational boundaries (input) | yes | yes |
| RM 1 | Operational boundaries (output) | yes | yes |
| RM 1 | Linear input–output relationship | yes | yes |
| RM 1 | System state selection | yes | yes |
| RM 1 | State sequences | yes | yes |
| RM 1 | Holding durations | yes | yes |
| RM 1 | Ramp limits | yes | yes |
| RM 2 | Operational boundaries (input) | yes | yes |
| RM 2 | Operational boundaries (output) | yes | yes |
| RM 2 | Linear input–output relationship | yes | yes |
| RM 2 | System state selection | yes | yes |
| RM 2 | State sequences | yes | yes |
| RM 2 | Holding durations | yes | yes |
| RM 2 | Ramp limits | yes | yes |
| System | Correlative dependency Input—{RM1, RM2} | yes | yes |
| System | Correlative dependency {RM1, RM2}—output | yes | yes |

Inst.—Instantiated. Para.—Parameterized.

The optimized schedule is shown in Figure 10. Therein, Figure 10a shows the electric power input of both refrigeration machines and Figure 10b shows the thermal power output. As described in the Section 4.2.3, offline analysis shows that all constraints of the optimization model are satisfied.

A comparison of the coefficients of performance (COPs) of refrigeration machines 1 and 2 shows that refrigeration machine 2 ($COP_{RM2}$: 2.7 vs. $COP_{RM1}$: 3.2) is less efficient. This is not reflected in the optimized schedule shown in Figure 10, as identical targets were set.

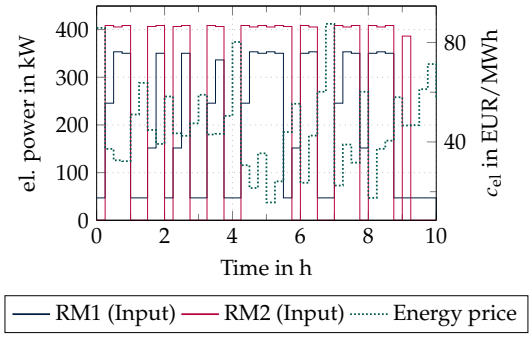
To achieve more efficient resource utilization, a combined target for both resources could be set, setting a target of 12,000 kWh of thermal energy instead of separate targets of 6000 kWh for each refrigeration machine.

Even though targets on the system level are not part of the model structure (Section 3.2), the extension of the model instance to incorporate use case-specific constraints, such as a target for specific flows, further demonstrates the user friendliness of the model generation methodology. The constraint for this target $T$ is shown in Equation (3) and is analogously modeled as shown in Equation (A6).

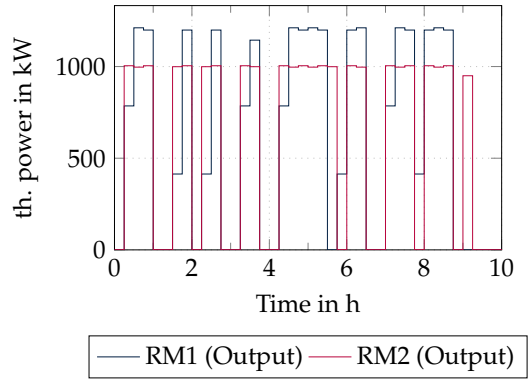
$$\Delta t \cdot \sum_{t \in |\mathcal{T}|} P_{\mathrm{RM1,\,out},t} + P_{\mathrm{RM2,\,out},t} = T \tag{3}$$

The resulting optimized schedule is shown in Figure 11. Compared to the schedule with equivalent targets for both refrigeration machines (Figure 10), this schedule operates the more efficient refrigeration machine 1 more frequently than refrigeration machine 2. Specifically, as shown in Figure 11a, refrigeration machine 1 generated 10,354 kWh, whereas refrigeration machine 2 produced only 1646 kWh of thermal energy, primarily during periods of low electricity prices (see Figure 11b). Operating the system with this optimized schedule resulted in electric energy costs of EUR 135.16. Therefore, the costs for cooperative operation leveraging heterogeneous efficiencies are lower than those for standalone

operation (see Figure 10). The computational time for the generation of this optimized schedule was 0.18 s.

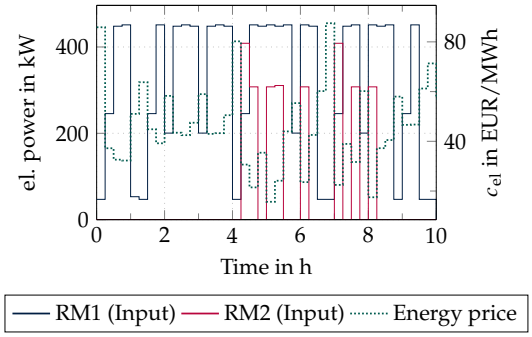



(**a**) Power Input

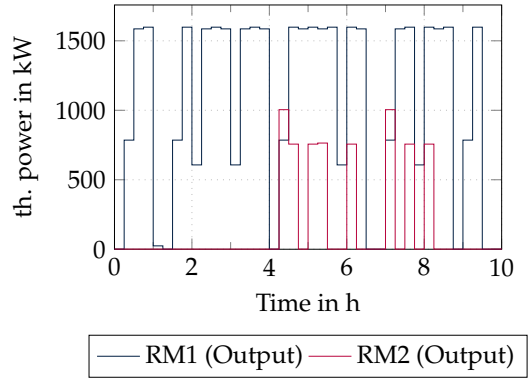



(**b**) Power Output

**Figure 10.** Optimized schedule for refrigeration system.




(**a**) Power Input




(**b**) Power Output

**Figure 11.** Optimized schedule for refrigeration system with target for system output.

### *4.4. Summary of the Evaluation*

The case studies presented in Sections 4.2 and 4.3 demonstrate the applicability of the methodology presented in Section 3 to different kinds of systems of flexible energy resources.

The parameters derived using Algorithm 2 show very good alignment with resource behavior (nRMSE values of 1–6%).

The verification of the generated model instances show that all expected elements of the model structure have been instantiated and subsequently parameterized by applying Algorithm 3. Eventually, the feasibility of the models for the generation of optimized schedules underlines the applicability of the methodology to the generation of usable optimization models.

The optimization case studies in Sections 4.2.3 and 4.3.3 highlight the economic benefits of energy-flexible resource operation, demonstrating that flexible operation is financially more advantageous than constant operation. Furthermore, the case study on the refrigeration system demonstrates the usefulness of aggregating resources within a system and the integrated planning of resource operation compared to the parallel planning of resource operation.

In summary, the evaluation shows the applicability of Algorithm 1. The optimization models for the CHP and the refrigeration system were generated without requiring manual involvement in the model creation process. This automation significantly accelerates the model generation process and reduces the potential for errors. In contrast, manual modeling is an error-prone, time-consuming process that relies heavily on the expertise of the practitioner [3,37]. As described in Section 1, the uptake of energy-flexible resource operation needs to be accelerated in the face of increasing shares of volatile renewable energy generation. This can only be achieved when domain knowledge in optimization modeling is not necessary for the creation of optimization models for systems of flexible energy resources.

As described in Section 1, MILP models are favored for the operational planning of flexible energy resources. Building upon this model type, reducing the complexity of the models is crucial [48] and thus is the focus during the generation process, e.g., when selecting an appropriate temporal resolution for the model [23]. This ensures that the generated optimization models are practical for near real-time operations, where short computational times are necessary. This is underlined by the short computational times of both case studies. Such efficiency allows the models to be used for real-time optimization to adapt resource operation swiftly to dynamic conditions, like changing forecasts in wind power. Henkel et al. [38] further demonstrate the usability of the model structure (Section 3.2) for the dual-use of an optimization model in both site-wide optimization (resource operation planning for durations of approx. 1 day, as shown in Sections 4.2 and 4.3) and real-time optimization. Thus, the automatically generated optimization models can also be applied for real-time optimization to incorporate volatile, hard-to-forecast renewable energy, such as wind power.

As described in Section 1, the traceability and comprehensibility of the model instances must be ensured (Requirement ④). The model structure and its parameters represent the physical behavior of resources: operational boundaries and input–output relationships correspond to recognizable limits of resources. System states and corresponding parameters are also identifiable by resource operators, as they align with resource properties. Dependencies within the system represent connections between resources. Validating these elements does not require an in-depth understanding of optimization modeling but is feasible with knowledge of the system in question. Thus, the model's traceability and comprehensibility are ensured. Additionally, as outlined in Section 3.3.2 and demonstrated

in Section 4.3.3, adapting specific model elements is easily feasible due to the modular model structure and the sequential model generation process.

In comparison with related work analyzed in Section 2, this work presents a methodology for the *automatic* generation of optimization models, covering a wider range of flexibility features and a comprehensive methodology for the automatic derivation of parameters. These two key components are then leveraged to facilitate the fully automated generation of models. As concluded in Section 2, existing approaches currently only focus on a subset of constraints and most often do not present methods for complete parameterization of models of systems or for the complete automation of model generation, thus limiting the ability of the methods to automatically create models.

## 5. Discussion

The methodology introduced in this work effectively minimizes manual intervention by automatically generating accurate optimization models, making the process more accessible and error-free. It can accommodate various types of flexible energy resources using a modular optimization model structure and sequential generation process, enhancing traceability and model management. This automation streamlines the optimization of energy resources, expediting the integration of renewable energy and contributing to energy system resilience and reduced operational costs.

The systematic review by Wagner et al. [9] of modeling approaches for flexible energy resources indicates that resources involving thermal dynamics typically require higher levels of detail in their models to adequately capture these dynamics. Consequently, using the model structure depicted in Figure 2 for resources influenced by thermal dynamics might lead to discrepancies between the model and actual resource behavior. The underlying optimization model structure draws on an analysis of existing generic models [3] and an examination of the required level of detail for optimization models [48]. Thus, this structure reflects a consensus within the scientific community and strives to balance modeling accuracy with computational efficiency. Despite potential deviations, these are within the acceptable range established by current standards [49]. Additionally, research by Reinpold et al. [37] demonstrates that even complex resources, such as distillation units, can be effectively represented by this model structure, highlighting its versatility and robustness. Therefore, despite the relatively simple model structure serving as the foundation for the automatically generated optimization model, the model sufficiently captures the operational behavior for the purposes of operational planning, such as in the context of energy markets, as long as the discrepancies between model predictions and actual performance, measured by the nRMSE [28], remain within acceptable margins. Moreover, the suitability of using simple models for such applications was confirmed by the findings of Wagner et al. [9], who based their conclusion on an analysis of 674 models of flexible energy resources.

Another limitation is the dependency on the operational behavior captured within time series datasets. The parameter set and thus the optimization model itself only reflects the operational behavior present in the data, necessitating comprehensive datasets that cover a wide array of operational scenarios to enhance model robustness. This limitation underscores the importance of utilizing a representative data set.

The computational complexity of MILP models scales close to exponentially [13]. Thus, optimizing large systems with a high number of flexible energy resources can be computationally challenging. However, optimization models for larger systems can be solved quickly and in time for their conversion into setpoints, i.e., between the disclosure of energy prices and the first time step of the optimization horizon, on consumer-grade

computers [38,48]. Thus, leveraging more powerful computing technology is a feasible way of ensuring timely generated schedules.

User trust in the automated process is also critical. To address this, the resulting optimization models are made traceable and understandable through a step-wise generation approach. Since the underlying data model for model parameters is constructed in such a way that parameters reflect the real behavior of the system, trust is fostered through transparency. This is pivotal, as users need to rely on the models for decision-making in real operational settings.

While the methodology advances the field significantly, it does not fully bridge the gap in domain knowledge necessary for operating energy resources flexibly, which might limit the use of generated models. This area, while outside the current scope of this research, highlights the potential for future advancements.

## 6. Conclusions

This work introduces a novel methodology for the automatic generation of optimization models for systems of flexible energy resources, significantly reducing the need for manual intervention. The methodology employs a generic optimization model structure to accommodate diverse systems of flexible energy resources. Parameters are automatically derived from time series data and a standardized information model of the system's structure, which are then used to create a parameterized optimization model for the generation of an optimized schedule for subsequent control of the system.

Two case studies—one with a CHP system and another involving multiple refrigeration machines of a refrigeration system—illustrate the methodology's broad applicability and effectiveness. These studies confirm the accuracy of the derived parameters, the correct instantiation and parameterization of constraints, and the feasibility of the generated optimization models.

By automating the model generation process, the methodology addresses significant challenges such as "competence obstacles" [15] and the intricacies of model development [16–18]. Moreover, the automated process fosters understanding and trust among users, which are critical for adopting such advanced models in practical energy management scenarios.

Future work could focus on enhancing the method's adaptability and user-friendliness. Developing a catalogue of objective functions will allow users to tailor models to specific environmental or operational goals, such as minimizing carbon dioxide emissions or optimizing load balancing. Additionally, integrating this methodology into existing optimization frameworks like *pycity_scheduling* [50] could further streamline its adoption and maximize its impact across the energy sector. The proposed methodology could also be applied to larger systems of heterogeneous energy resources.

**Author Contributions:** L.P.W.: conceptualization, data curation, investigation, methodology, validation, visualization, writing—original draft, writing—review and editing. F.G.: supervision, writing—review and editing. L.M.R., G.F. and J.J.: writing—review and editing. A.F.: funding acquisition, supervision, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research, within the project OptiFlex, was funded by dtec.bw—Digitalization and Technology Research Center of the Bundeswehr. dtec.bw is funded by the European Union—NextGenerationEU.

**Data Availability Statement:** The implementation and the results are available at https://github.com/lukas-wagner/AutoModelGeneration, accessed on 6 January 2025.

**Acknowledgments:** The authors thank Cirera et al. [46] for sharing their dataset of the operation of the refrigeration system.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this work:

| | |
|---|---|
| **CHP** | combined heat and power |
| **COP** | coefficient of performance |
| **FPD** | formalized process description |
| **MILP** | mixed-integer linear programming |
| **nRMSE** | normalized root mean square error |
| **PLA** | piecewise linear approximation |
| **$R^2$** | coefficient of determination |
| **SOC** | state of charge |

## Appendix A

*Appendix A.1 Constraints Encapsulated Within Flexibility Features*

The constraints encapsulated within the flexibility features described in this section are reproduced from Wagner et al. [3].

*Operational boundaries* are modeled as shown in Equation (A1) [3].

$$P_{\min} \leq P_t \leq P_{\max} \qquad \forall t \tag{A1}$$

The piecewise linear approximation of the *input–output relationship* is realized using binary variables $x_k$ for each segment $k \in \mathcal{K}$ and time step $t$ (Equations (A2)–(A4). Only one segment can be active per time step (Equation (A5)). For a linearized representation of the *input–output relationship*, Equations (A3)–(A5) are omitted and $|\mathcal{K}| = 1$ in Equation (A2). The total energy output (target) $D$ over the optimization horizon is set by Equation (A6). Depending on whether a target for the input or the output of the resource is to be set, $P_t$ in Equation (A6) corresponds to the respective decision variable, i.e., either $P_{\text{input}_k,t}$ or $P_{\text{output},t}$ [3].

$$P_{\text{output},t} = \sum_{k \in \mathcal{K}} \left( a_k \cdot P_{\text{input}_k,t} + b_k \cdot x_{k,t} \right) \qquad \forall t \tag{A2}$$

$$\text{lb}_k \cdot x_{k,t} \leq P_{\text{input}_k,t} \qquad \forall t, k \tag{A3}$$

$$P_{\text{input}_k,t} \leq \text{ub}_k \cdot x_{k,t} \qquad \forall t, k \tag{A4}$$

$$\sum_{k \in \mathcal{K}} x_{k,t} = 1 \qquad \forall t \tag{A5}$$

$$\Delta t \cdot \sum_{t \in \mathcal{T}} P_t = D \tag{A6}$$

*System states* $s \in \mathcal{S}$ are characterized by lower and upper flow limits (Equations (A8) and (A9)), follower states $\mathcal{S}_{F,s}$ (Equation (A10)), holding durations of each state $s$ (Equations (A11) and (A12)), and ramp limits (Equations (A13) and (A14)) [3].

$$\sum_{s \in \mathcal{S}} x_{s,t} = 1 \qquad \forall t \tag{A7}$$

$$P_t \geq \sum_{s \in \mathcal{S}} P_{\min,s} \cdot x_{s,t} \qquad \forall t > t_0 \tag{A8}$$

$$P_t \leq \sum_{s \in \mathcal{S}} P_{\max,s} \cdot x_{s,t} \qquad \forall t > t_0 \tag{A9}$$

$$x_{t-1,s} - x_{t,s} \leq \sum_{f \in \mathcal{S}_{F,s}} x_{f,t} \qquad \forall s, t > t_0 \tag{A10}$$

$$t_{h,\min,s} \cdot \left(x_{t,s} - x_{t-1,s}\right) \leq \sum_{\tau \in \mathcal{T}_h} x_{\tau,s} \qquad \forall s, t > t_0 \tag{A11}$$

$$t_{h,\max,s} \geq \sum_{\tau \in \mathcal{T}_h} x_{\tau,s} \qquad \forall s, t > t_0 \tag{A12}$$

$$\Delta t \cdot \sum_{s \in \mathcal{S}} \left(\text{ramp}_{\min,s} \cdot x_{s,t}\right) \leq |P_t - P_{t-1}| \qquad \forall t > t_0 \tag{A13}$$

$$\Delta t \cdot \sum_{s \in \mathcal{S}} \left(\text{ramp}_{\max,s} \cdot x_{s,t}\right) \geq |P_t - P_{t-1}| \qquad \forall t > t_0 \tag{A14}$$

The *correlative dependency* relationship of resources within the system is modeled as shown in Equation (A15). Therein, the direction of the flows involved in one dependency relation is retained through the use of the respective decision variables corresponding to the input and output flows of the resources. This is done per energy carrier $e$. One resource cannot be involved on both input and output sides simultaneously ($r_k \neq r_j$). [3]

$$\sum_{r_k \in \mathcal{R}_k} P_{\text{output},r_k,e,t} = \sum_{r_j \in \mathcal{R}_j} P_{\text{input},r_j,e,t} \qquad \forall t, e, r_k \neq r_j \tag{A15}$$

For *restrictive dependencies*, the constraints are modeled as shown in Equations (A16)–(A20). Equations (A16) and (A17) ensure that only one flow of one resource is active per side of the dependency through the use of binary decision variables. Equations (A18)–(A20) connect the active flows of the dependency relation by intermediate decision variables $P_{\text{input},t}$ and $P_{\text{output},t}$ [3].

$$\sum_{r_k \in \mathcal{R}_k} x_{r_k,e,t} = 1 \qquad \forall e, t \tag{A16}$$

$$\sum_{r_j \in \mathcal{R}_j} x_{r_j,e,t} = 1 \qquad \forall e, t \tag{A17}$$

$$P_{\text{output},r_k,t} = P_{\text{output},t}, \qquad \text{if } x_{r_k,t} = 1 \quad \forall r_i, t \tag{A18}$$

$$P_{\text{input},r_j,t} = P_{\text{input},t}, \qquad \text{if } x_{r_j,t} = 1 \quad \forall r_j, t \tag{A19}$$

$$P_{\text{output},e,t} = P_{\text{input},e,t} \qquad \forall e, t \tag{A20}$$

## References


1. Fachot, M. Moving to An All-Electric Society. 2022. Available online: https://etech.iec.ch/issue/2022-05/moving-to-an-all-electric-society (accessed on 6 January 2025).
2. Ulbig, A.; Andersson, G. On operational flexibility in power systems. In Proceedings of the 2012 IEEE Power and Energy Society General Meeting, San Diego, CA, USA, 22–26 July 2012; pp. 1–8. [CrossRef]
3. Wagner, L.P.; Reinpold, L.M.; Fay, A. Design Patterns for Optimization Models of Flexible Energy Resources. In Proceedings of the 2nd IEEE Industrial Electronics Society Annual Online Conference (ONCON), Online, 8–10 December 2023; pp. 1–6. [CrossRef]
4. *ISO 50003:2021-05*; Energy Management Systems—Requirements for Bodies Providing Audit and Certification of Energy Management Systems. DIN Media: Berlin, Germany, 2021. [CrossRef]
5. Misconel, S.; Leisen, R.; Mikurda, J.; Zimmermann, F.; Fraunholz, C.; Fichtner, W.; Möst, D.; Weber, C. Systematic comparison of high-resolution electricity system modeling approaches focusing on investment, dispatch and generation adequacy. *Renew. Sustain. Energy Rev.* **2022**, *153*, 111785. [CrossRef]
6. Johnsen, A.G.; Mitridati, L.; Zarrilli, D.; Kazempour, J. The Value of Ancillary Services for Electrolyzers. *arXiv* **2023**, arXiv:2310.04321.
7. Reinpold, L.M.; Wagner, L.P.; Kilthau, M.; Karmann, A.; Fay, A. Planning functions for (Energy) Storage Systems. *Atp Mag.* **2023**, *65*, 60–69. [CrossRef]
8. Bundesnetzagentur. *Key Points Paper on the Further Development of Industrial Grid Charges in the Electricity Sector. (Eckpunktepapier zur Fortentwicklung der Industrienetzentgelte im Elektrizitätsbereich)*; Bundesnetzagentur: Bonn, Germany, 2024.
9. Wagner, L.P.; Reinpold, L.M.; Kilthau, M.; Fay, A. A Systematic Review of Modeling Approaches for Flexible Energy Resources. *Renew. Sustain. Energy Rev.* **2023**, *184*, 113541. [CrossRef]
10. Tadayon, L.; Meiers, J.; Ibing, L.; Erdelkamp, K.; Frey, G. Coordinated Dispatch of Battery and Pumped Hydro Energy Storage utilizing hydraulic short circuit. In Proceedings of the 2024 9th International Youth Conference on Energy (IYCE), Colmar, France, 2–6 July 2024; pp. 1–7. [CrossRef]
11. Riaz, S.; Mancarella, P. Modelling and Characterisation of Flexibility From Distributed Energy Resources. *IEEE Trans. Power Syst.* **2022**, *37*, 38–50. [CrossRef]
12. Ommen, T.; Markussen, W.B.; Elmegaard, B. Comparison of linear, mixed integer and non-linear programming methods in energy system dispatch modelling. *Energy* **2014**, *74*, 109–118. [CrossRef]
13. Richards, A.; How, J. Mixed-integer programming for control. In Proceedings of the 2005 American Control Conference, Portland, OR, USA, 8–10 June 2005; pp. 2676–2683. [CrossRef]
14. Baumhof, M.T.; Raheli, E.; Johnsen, A.G.; Kazempour, J. Optimization of Hybrid Power Plants: When is a Detailed Electrolyzer Model Necessary? In Proceedings of the 2023 IEEE PowerTech, Belgrade, Serbia, 25–29 June 2023; pp. 1–10. [CrossRef]
15. Leinauer, C.; Schott, P.; Fridgen, G.; Keller, R.; Ollig, P.; Weibelzahl, M. Obstacles to demand response: Why industrial companies do not adapt their power consumption to volatile power generation. *Energy Policy* **2022**, *165*, 112876. [CrossRef]
16. Krishnamoorthy, D.; Skogestad, S. Real-Time optimization as a feedback control problem—A review. *Comput. Chem. Eng.* **2022**, *161*, 107723. [CrossRef]
17. Allen, N.A.; Shaffer, C.A.; Watson, L.T. Building Modeling Tools That Support Verification, Validation, and Testing for the Domain Expert. In Proceedings of the Winter Simulation Conference, Orlando, FL. USA, 4–7 December 2005; pp. 419–426. [CrossRef]
18. Farooqui, A.; Falkman, P.; Fabian, M. Towards Automatic Learning of Discrete-Event Models from Simulations. In Proceedings of the 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), Munich, Germany, 20–24 August 2018; pp. 857–862. [CrossRef]
19. Henkel, V.; Wagner, L.P.; Kilthau, M.; Gehlhoff, F.; Fay, A. A Multi-Agent Approach for the Optimized Operation of Modular Electrolysis Plants. *Energies* **2024**, *17*, 3370. [CrossRef]
20. Chicco, G.; Mancarella, P. Matrix modelling of small-scale trigeneration systems and application to operational optimization. *Energy* **2009**, *34*, 261–273. [CrossRef]
21. Wang, Y.; Cheng, J.; Zhang, N.; Kang, C. Automatic and linearized modeling of energy hub and its flexibility analysis. *Appl. Energy* **2018**, *211*, 705–714. [CrossRef]
22. Ramonat, M.; Fay, A. Method for the parameterization of simulation models by the use of plant measurement data (ger.: Methode zur Parametrierung von Simulationsmodellen durch Nutzung von Anlagenmessdaten). In Proceedings of the 24. VDI-Kongress AUTOMATION, Baden-Baden, Germany, 27–28 June 2023; pp. 167–178. [CrossRef]
23. Wagner, L.P.; Fay, A. Methodology for Deriving Parameters for Optimization Models of Systems of Flexible Energy Resources. *IEEE Open J. Ind. Electron. Soc.* **2024**, *5*, 737–757. [CrossRef]
24. Barth, M.; Fay, A. Automated generation of simulation models for control code tests. *Control Eng. Pract.* **2013**, *21*, 218–230. [CrossRef]

25. Novák, P.; Ekaputra, F.J.; Biffl, S. Generation of Simulation Models in MATLAB-Simulink Based on AutomationML Plant Description. *IFAC-PapersOnLine* **2017**, *50*, 7613–7620. [CrossRef]
26. Ramonat, M.; Kunze, F.C.; Gehlhoff, F.; Fay, A. Identifying Root-Causes of Deviations between Simulation and Real Plant Data based on an Adaptive Causal Directed Graph. In Proceedings of the 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), Padova, Italy, 10–13 September 2024; pp. 1–8. [CrossRef]
27. Vorm, E.S.; Combs, D.J.Y. Integrating Transparency, Trust, and Acceptance: The Intelligent Systems Technology Acceptance Model (ISTAM). *Int. J. Hum.-Interact.* **2022**, *38*, 1828–1845. [CrossRef]
28. Wen, X.; Jaxa-Rozen, M.; Trutnevyte, E. Accuracy indicators for evaluating retrospective performance of energy system models. *Appl. Energy* **2022**, *325*, 119906. [CrossRef]
29. Panagoulias, D.P.; Sarmas, E.; Marinakis, V.; Virvou, M.; Tsihrintzis, G.A.; Doukas, H. Intelligent Decision Support for Energy Management: A Methodology for Tailored Explainability of Artificial Intelligence Analytics. *Electronics* **2023**, *12*, 4430. [CrossRef]
30. Kasper, L.; Schwarzmayr, P.; Birkelbach, F.; Javernik, F.; Schwaiger, M.; Hofmann, R. A digital twin-based adaptive optimization approach applied to waste heat recovery in green steel production: Development and experimental investigation. *Appl. Energy* **2024**, *353*, 122192. [CrossRef]
31. Förderer, K.; Ahrens, M.; Bao, K.; Mauser, I.; Schmeck, H. Modeling flexibility using artificial neural networks. *Energy Inform.* **2018**, *1*, 21. [CrossRef]
32. Manna, C.; Lahariya, M.; Karami, F.; Develder, C. A data-driven optimization framework for industrial demand-side flexibility. *Energy* **2023**, *278*, 127737. [CrossRef]
33. Ramonat, M.; Fay, A. Method for Automatic Simulation Model Calibration and Maintenance for Brownfield Process Plants. In Proceedings of the 32nd International Symposium on Industrial Electronics (ISIE), Helsinki, Finland, 19–21 June 2023; pp. 1–6. [CrossRef]
34. Martinez, G.S.; Sierla, S.; Karhela, T.; Vyatkin, V. Automatic Generation of a Simulation-Based Digital Twin of an Industrial Process Plant. In Proceedings of the IECON 2018—44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 21–23 October 2018; pp. 3084–3089. [CrossRef]
35. Sierla, S.; Sorsamäki, L.; Azangoo, M.; Villberg, A.; Hytönen, E.; Vyatkin, V. Towards Semi-Automatic Generation of a Steady State Digital Twin of a Brownfield Process Plant. *Appl. Sci.* **2020**, *10*, 6959. [CrossRef]
36. Wagner, L.P.; Reinpold, L.M.; Kilthau, M.; Gehlhoff, F.; Derksen, C.; Loose, N.; Jepsen, J.; Fay, A. Utilizing Mass Storage for Flexibilizing Energy Resource Operation: Cost-Efficient Resource Operation by Responding to Market Prices. *Atp Mag.* 2025, *accepted*. [CrossRef]
37. Reinpold, L.M.; Wagner, L.P.; Reiche, L.T.; Fay, A. Experimental Setup for the Evaluation of Optimization Strategies for Flexible Energy Resources. In Proceedings of the 2nd IEEE Industrial Electronics Society Annual Online Conference (ONCON), Online, 8–10 December 2023; pp. 1–6. [CrossRef]
38. Henkel, V.; Wagner, L.P.; Gehlhoff, F.; Fay, A. Combination of Site-Wide and Real-Time Optimization for the Control of Systems of Electrolyzers. *Energies* **2024**, *17*, 4396. [CrossRef]
39. Wanapinit, N.; Thomsen, J.; Kost, C.; Weidlich, A. An MILP model for evaluating the optimal operation and flexibility potential of end-users. *Appl. Energy* **2021**, *282*, 116183. [CrossRef]
40. Barth, L.F.J.; Ludwig, N.N.; Mengelkamp, E.; Staudt, P. A comprehensive modelling framework for demand side flexibility in smart grids. *Comput. Sci.-Res. Dev.* **2018**, *33*, 13–23. [CrossRef]
41. Ahčin, P.; Šikić, M. Simulating demand response and energy storage in energy distribution systems. In Proceedings of the 2010 International Conference on Power System Technology, Zhejiang, China, 24–28 October 2010; pp. 1–7. [CrossRef]
42. Verlag des Vereins Deutscher Ingenieure. *VDI/VDE 3682 Part 1: 2015-05: Formalised Process Descriptions—Concept and Graphic Representation*; Verlag des Vereins Deutscher Ingenieure: Düsseldorf, Germany, 2015.
43. IBM. *IBM ILOG CPLEX Optimizer (V22.1.0)*; IBM: Armonk, NY, USA, 2022.
44. Nabizada, H.; Köcher, A.; Hildebrandt, C.; Fay, A. Open, web-based tool for information modeling with formalized process description (ger.: Offenes, webbasiertes Werkzeug zur Informationsmodellierung mit Formalisierter Prozessbeschreibung). In Proceedings of the 21 VDI-Kongress AUTOMATION, Baden-Baden, Germany, 30 June–1 July 2020; pp. 443–454. [CrossRef]
45. Continuous Intra-Day Prices for Germany, 2024. Available online: www.epexspot.com (accessed on 6 January 2025).
46. Cirera, J.; Pujal, M.; Brull, J.; Vendrell, A. Industrial Overfeed Refrigeration System. 2020. Available online: https://ieee-dataport.org/documents/industrial-overfeed-refrigeration-system (accessed on 6 January 2025).
47. Cirera, J.; Carino, J.A.; Zurita, D.; Ortega, J.A. Improving the Energy Efficiency of Industrial Refrigeration Systems by Means of Data-Driven Load Management. *Processes* **2020**, *8*, 1106. [CrossRef]
48. Wagner, L.P.; Kilthau, M.; Reinpold, L.M.; Fay, A. Required Level of Detail of Optimization Models for the Control of Flexible Energy Resources. In Proceedings of the 2023 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, Glasgow, UK, 31 October–3 November 2023; pp. 1–6. [CrossRef]

49. Raheli, E.; Werner, Y.; Kazempour, J. Flexibility of Integrated Power and Gas Systems: Gas Flow Modeling and Solution Choices Matter. *IEEE Trans. Power Syst.* **2024**, 1–13. [CrossRef]
50. Schwarz, S.; Uerlich, S.A.; Monti, A. pycity_scheduling—A Python framework for the development and assessment of optimisation-based power scheduling algorithms for multi-energy systems in city districts. *SoftwareX* **2021**, *16*, 100839. [CrossRef]