IEEE Open Journal of the
Industrial Electronics Society

Received 7 May 2024; accepted 6 July 2024. Date of publication 10 July 2024;
date of current version 25 July 2024. The review of this article was arranged by Associate Editor Jin-Gang Lai.

Digital Object Identifier 10.1109/OJIES.2024.3425934

Methodology for Deriving Parameters for Optimization Models of Systems of Flexible Energy Resources

LUKAS PETER WAGNER [1] AND ALEXANDER FAY [2] (Senior Member, IEEE)

[1]Institute of Automation Technology, Helmut Schmidt University, 22043 Hamburg, Germany
[2]Chair of Automation, Ruhr University, 44801 Bochum, Germany

CORRESPONDING AUTHOR: LUKAS PETER WAGNER (e-mail: lukas.wagner@hsu-hh.de).

This work in the project OptiFlex was funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr and dtec.bw is funded by the European Union – NextGenerationEU.

ABSTRACT The increasing share of renewable energy generation poses a challenge to maintaining the adequacy of power generation and demand. Planning the operation of flexible energy resources helps stabilize this balance. Optimization models are needed for operation planning. The use of a predefined and validated optimization model avoids modeling errors, but parameterization is still necessary and error-prone. This work presents a methodology for determining parameters of optimization models for various kinds of flexible energy resources. The parameters are derived from time series data of the resource operation after preprocessing. This methodology includes algorithms for determining operational boundaries, the input–output relationship, system states, and parameters for storage systems. Connections of flows between individual resources within a system are extracted from a standardized information model. A case study of a combined heat and power system demonstrates the applicability of the methodology by deriving a set of parameters from time series data and an information model of the system's structure. The model parameterized by means of the methodology shows very good alignment with a validation time series data set with a normalized root mean square error of 1% (generator), respectively, of 6% (heat exchanger).

INDEX TERMS Energy flexibility, parameter derivation, optimization model.

I. INTRODUCTION

It is anticipated that by 2030, intermittent renewable energy sources, such as wind and solar, will generate 60% of the European Union's renewable electricity [1]. This necessitates the corresponding development or enablement of flexible power generation and demand [2]. Thus, energy flexibility gains momentum in short-term resource operation planning [3]. Fluctuating prices in energy markets further incentivize the use of energy flexibility [4]. *Energy flexibility* is commonly referred to as the ability of flexible energy resources, such as generators, energy storage systems, or consumers, to modulate their power input or output [5].

A literature review conducted in [6] shows that the planning of the operation of various common types of flexible energy resources is often achieved by means of mixed-integer linear programming (MILP) optimization. It is also shown, that many optimization models for the planning of resource operation exist, but their reuse is rare [6]. To achieve reuse ability, design patterns for instantiating MILP optimization models of flexible energy resources and systems thereof are presented in [7]. The use of design patterns greatly reduces the modeling and implementation effort. Potential modeling errors are avoided due to the preexisting, already validated model structure. However, the parameterization of the optimization model still requires manual effort and knowledge of a domain expert and can lead to inaccurate results or infeasibility of the optimization model [7]. In addition, the accuracy of the optimized schedule heavily depends on the model used, i.e., on the model structure but also on the set of parameters [8].

Furthermore, the state-of-the-art industrial information technology enables the logging of energy related data from heterogeneous resources [9].

Therefore, this work investigates within research question 1 how to reduce the parameterization effort of a generic MILP optimization model, following the modeling approach detailed in Section II, by means of automatic derivation of numerical parameters from time series data of the resource operation. To achieve this research objective, the following requirements ⓗ must be met. To make the parameters accessible for the parameterization of the optimization model, an appropriate data model is required ① [10], [11]. Deriving numerical parameters from data requires a method for data preprocessing, consisting of statistical analyses of the suitability of the data ②, including a check for the significance of the relationship of different time series data columns [12], [13] as well as identifying and replacing erroneous values ③ [14], [15], [16] to ensure meaningful parameter derivation.

Parameters required for the optimization are operational boundaries ④, coupling parameters of input and output flows, i.e., the input–output relationship ⑤, and system states and related parameters ⑥, such as state boundaries, state sequences, holding durations, and ramp limits [7]. In addition to these parameters, storage systems ⑦ require additional parameters to account for their conversion efficiencies [7].

The relevant data for this work are time series data of the resource operation. A discrete time series $\mathcal{X}$ is a sequence of values x_t (with $x_t \in \mathbb{R}$) with length $|\mathcal{T}| \in \mathbb{N}$. Each value is associated with a time stamp $t \in \mathcal{T}$. The operation of a flexible energy resource is described by multiple time series, henceforth called "time series data columns." Individual values are measured simultaneously, i.e., have the same time stamp. In addition, metadata, such as name of the time series data and unit of the measurement, are of interest for the correct assignment during the derivation of parameters.

Individual flexible energy resources exhibit so called "dependencies," which are connections of flows of resources within a system of resources that must be accounted during the optimization. A dependency relationship is characterized by one list each of outputs of resources and inputs of resources involved. The type of a dependency of resources is either correlative, i.e., any flows of the resources involved can be active, or restrictive, i.e., exactly two resources can be connected by a flow per dependency and time step of the optimization [7].

Hence, this work also investigates within research question 2 how dependencies of flexible energy resources can be extracted from an information model representing the connections of resources within a system to parameterize an optimization model of a system of flexible energy resources. Further requirements must be fulfilled for this research objective. The method must allow for a complete extraction of the dependencies including the explicit assignment of the resources involved to each dependency relation ⑧ [17], [18] and their type ⑨ [18], [19]. The data model must also include dependency-related parameters ① [10].

Information modeling is a common approach to representing information in a machine-readable form that can also be used for the exchange of information between applications [20], [21]. There are multiple widely used standards for information modeling of processes, such as AutomationML [20] or the formalized process description (FPD) [21]. These standards [20], [21] allow for the modeling of the system's structure [22]. Often, information models of the system structure already exist [23].

To answer the research questions, this work provides the following contributions:

1) development of a data model for the persistence of derived parameters and dependency-related information;
2) integrated method for the automated preprocessing of time series data and the derivation of numerical parameters for optimization models of systems of flexible energy resources from time series data of resource operation;
3) method for the extraction of dependency-related information from an FPD information model.

The rest of this article is organized as follows. Section II describes the underlying modeling approach for systems of flexible energy resources. Section III describes the related work as well as the respective fulfillment of the requirements. Section IV presents the methodology of the derivation of numerical parameters and extraction of dependency-related information. Section V describes the validation by means of a case study. Section VI discusses the methodology. Finally, Section VII concludes this article and describes future work.

II. MODELING OF SYSTEMS OF FLEXIBLE ENERGY RESOURCES

This section describes the underlying modeling approach for systems of flexible energy resources and infers which parameters are required. The parameters necessary correspond to requirements ④–⑨.

The design patterns for optimization models presented by Wagner et al. [7] allow for the modeling of flexible energy resources and systems thereof by means of selecting the necessary patterns to represent the resource behavior. The mathematical model has been created after an analysis of related work in generic MILP optimization models for flexible energy resources [7]. Fig. 1 shows a metamodel of the modeling approach presented in [7]. This model is the basis for the methodology presented in Section IV. For the description of the mathematical modeling of individual constraints, refer to [7].

As shown in Fig. 1, within one optimization model, one *system* consists of one to many *resources* [7]. The *system* is represented by one set of *operational boundaries* as well as the *resources*, which are connected by *dependencies* [7]. A *resource* can exhibit one to many *dependencies*, which are connections of the output flow of one resource to the input flow of another resource [7]. As defined in Section I, there are two types of *dependencies* of *resources:* correlative *dependencies* allow simultaneous flows from and to all participating *resources*, whereas restrictive *dependencies*, such as those in systems with switches, allow flows from/to only one participating *resource* per side and time step [7], [10]. Subsets

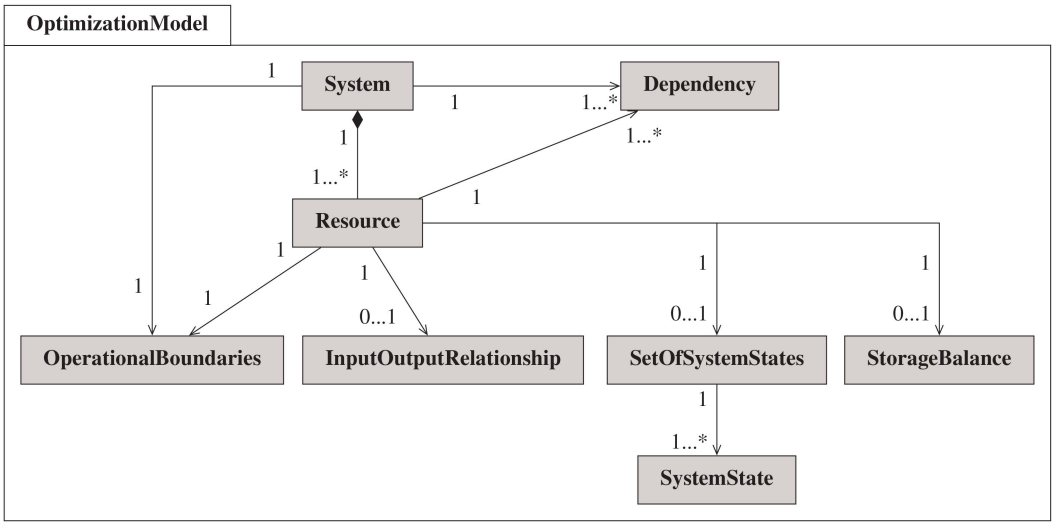


FIGURE 1. Metamodel of the modeling approach presented by Wagner et al. [7].

of the set of all *resources* are involved on both sides of the *dependency* [7].

Thus, the lists of *resources* that are part of a *dependency* on each side (input/output) ⑧ as well as the type ⑨ need to be determined.

A *resource* must also always exhibit *operational boundaries*. This includes the need for parameters for lower and upper bounds ④ [24]. These parameters are also necessary for the representation of the *operational boundaries* of the *system*, such as limits of the grid connection [7].

The behavior of a *resource* can be represented by either an *input–output relationship* or alternatively by a *storage balance*, if the *resource* exhibits some kind of storage. In both cases, the behavior of the *resource* can be further characterized by one *set of system states* [25].

The *input–output relationship* describes the relation of the output of a *resource* as a function of the input. In the case of a linearized representation, this can be achieved by means of a slope and intercept [7]. Alternatively, piecewise linear approximation (PLA) can be used to approximate the *input–output relationship* with linear segments consisting of slope, intercept, lower, and upper bounds. Thus, necessary parameters for an *input–output relationship* ⑤ are the slope and intercept or linear segments, consisting of slope, intercept, as well as lower and upper bounds [26].

Each *resource* can exhibit a *set of system states* or not. A *set of system states* consists of at least one *system state* that is defined by minimum and maximum limits for the decision variable corresponding to the input or the state of charge (SOC), in case of storage systems [25], [26]. There can be a set of allowable follower states, meaning that state 2 cannot be reached from state 0 without first passing through state 1 [25]. In addition, there are specific minimum or maximum holding durations that *system states* must adhere to [27]. It is also necessary to consider the allowable rate of change in flows, known as ramp limits, for each of the *system states* [27]. Therefore, parameters describing one *system state* ⑥ contain lower and upper bounds of the input flow or SOC, follower states, holding durations, and ramp limits.

Storage resource are modeled as a *storage balance*. This necessitates parameters for the capacity and conversion efficiencies ⑦ [24].

III. RELATED WORK

This section provides an analysis of related work in data models for energy flexibility (see Section III-A) and in numerical parameter derivation, including data preprocessing (Section III-B) and statistical analyses for the derivation of parameters (see Section III-C). It also analyzes related work in extracting dependency-related information from standardized information models (see Section III-D). A summary of the findings of this section is provided in Section III-E.

The respective fulfillment of each requirement ⓐ is represented by ●, partial fulfillment by ◐, and no fulfillment by ○ and is summarized in one table per research area. Requirements not applicable for one research area are marked with ■ in the table.

A. DATA MODEL

This section describes the related work in data models for flexible energy resources and the fulfillment of requirement ①.

A generic data model for the description of energy flexible loads, dependencies, and storage systems is described by Schott et al. [10]. The model includes parameters that describe "flexible load measures," which are predefined deviations from an operating schedule [10]. While this data model also aims to represent parameters of flexible energy resources and dependencies, the approaches of the flexible load-centric representation of energy flexibility and the corresponding data model of [10] are different from the modeling approach described in Section II, which focuses on representing the

TABLE 1. Fulfillment of Requirement for the Data Model

Requirement	①	② - ⑨
Schott *et al.* [10]	◐	■
Li et al.and Hong [28]	◐	■
Biegel et al. [30]	◐	■
IEC 61968-5 [32]	○	■
Reviewed in [7]	○	■

Fulfillment: ● – full ◐ – partial ○ – none ■ – n/a

flexibility of each resource rather than individual flexible load measures.

A semantic model for representing the energy flexibility of commonly used resources in buildings is presented by Li and Hong [28]. The semantic model is built based on an analysis of existing semantic models, such as the *DELTA* ontology [29], to instantiate a simulation model of the building by referencing objects from simulation applications, such as *TRNSYS* or *EnergyPlus*, for the respective resources. The model contains parameters for energy- and power-related boundaries of resources, but is neither designed to hold coupling parameters of input and output or a resource, system state related parameters, storage-related parameters, nor dependencies [28].

An information model for the control of distributed energy resources is presented by Biegel et al. [30]. The information model consists of different "flexibility blocks" for the description and communication of the distributed energy resources' flexibility to an aggregator. The model is able to represent a range of parameters, including operational boundaries and system state related parameters, such as ramp. Parameters for state sequences, holding durations, dependencies of individual resources, or the input–output relationship are not part of the model [30].

A series of standards within the "common information model" aim at describing various resources used for energy distribution for the standardized data exchange [31]. The standard family also contains multiple standards for the serialization [31]. Within this standard family, the standard [32] presents an interface description for the operational planning and optimization of distributed energy resources. The standard [32] includes parameters for the monitoring of individual resources, such as the unit of measurement, as well as Boolean parameters whether certain parameters are controllable, e.g., whether the voltage can be regulated [32]. Parameters describing resource dynamics, such as operational boundaries, input–output relationships, system state related parameters, or storage systems, are not included [32]. As this standard describes individual resources, dependencies are also not included [32].

The analysis of existing data models for energy flexibility is summarized in Table 1. It is shown that, although there exist different data or information models for the description of flexible energy resources [10], [28], [30], [32], none of them represents all the parameters necessary, as specified through requirements ④–⑨. Furthermore, the analysis of related work in [7] shows that none of the reviewed modeling approaches for flexible energy resources presents a corresponding data model.

B. DATA PREPROCESSING

This section describes related work in data preprocessing and the fulfillment of requirements ② and ③.

The work of Choi et al. [12] presents a method for developing a data-based, predictive model of a distillation process. The methodology includes a step of identifying relevant time series data columns by determining their correlation and stationarity ②. Stationarity means that the statistical properties of the data remain constant over time [33]. Cleaning the data, i.e., identifying and replacing outliers ③, is specifically not included in the method [12].

In the work of Luo et al. [13], load profiles of buildings are analyzed. Therein, the p-value is used to determine the significance of individual time series data columns. The p-value represents "the smallest significance level (...) at which the hypothesis [that no relationship exists] would be rejected for the given observation" [34]. This is done to decide which time series data columns to include in the analysis ②. Similar to the work in [12], only the identification of significant time series data columns is described but a method for the detection of erroneous values ③ is not presented [13].

A method for the elimination of outliers in time series data ③ of the operation of refrigeration machines is presented by Hechelmann [14]. Therein, a distinction is made between system behavior-based and statistical approaches. Since the derivation of numerical parameters is intended to be purely data-based in this work, no system behavior-based methods can be applied. Statistical preprocessing uses percentiles as thresholds, i.e., values below or above a defined percentile are excluded. However, removing erroneous values reduces the size of the dataset by 58%. An analysis of the correlation of different time series data columns is not conducted ② [14].

The work of Li et al. [15] focuses on the identification and characterization of patterns in the electrical load profiles of buildings and includes a step for identification of erroneous values ③. Similar to the work in [14], the results of the work in [15] show that "values that are more than three standard deviations away from the mean" could be considered as outliers, but the correlation of different time series data columns ② is not determined.

The work of Barbero et al. [16] presents a methodology for the data-based analysis of the energy flexibility potential of building-related energy resources. This includes a method for the handling of errors in time series data ③, such as missing values or outliers, and the application of linear regression to replace values, but not for determining correlations ② [16].

A method for the preprocessing of data is presented by Olariu et al. [35]. The method includes manual steps, such as "visual inspections." Linear interpolation is proposed for the handling of missing or erroneous values ③ [35].

TABLE 2. Fulfillment of Requirements for Data Preprocessing

Requirement	①	②	③	④ - ⑨
Choi *et al.* [12]	■	●	○	■
Luo *et al.* [13]	■	◐	○	■
Hechelmann [14]	■	○	◐	■
Li *et al.* [15]	■	○	◐	■
Barbero *et al.* [16]	■	○	●	Tab. 3
Olariu *et al.* [35]	■	○	◐	■

Fulfillment: ● – full ◐ – partial ○ – none ■ – n/a

Some works concentrate on the identification of correlations or the significance of time series data columns ② [12], [13]. Other works present methods for identifying or removing erroneous values ③ [14], [15], [16], or for the replacement of erroneous values ③ [16], [35]. None of these works applies an integrated method of identifying relevant time series data columns and the identification and replacement of erroneous values for the preprocessing. Thus, as listed in Table 2, only partial fulfillment of data preprocessing is seen.

C. PARAMETER DERIVATION

This section describes related work in parameter derivation from time series data of resource operation as well as the fulfillment of requirements ④–⑦.

Data-driven modeling, also called system identification [36], aims at determining parameters for mathematical models from "observed input–output data" [37]. According to a review conducted by Ljung [37], system identification stems from the need to "realize linear state-space models from impulse responses" [37], i.e., to parameterize first-principle models [36]. System identification is increasingly also applied for parameterizing gray box models for optimization [8], [38], [39], [40], [41], [42].

Parameters of a multiperiodic resistance–capacitance (*RC*)-equivalent state-space model of the dynamics of a thermostatically controlled load are fitted to data by [38] using an *RC*-equivalence model-specific statistical approach. The state-space model is then discretized for the use in MILP optimization, in the style of an energy balance ⑦ [38]. Similarly, the work of Reinpold et al. [39] presents the parameterization of a lumped thermal resistance simulation model of an experimental distillation unit as well as a corresponding MILP optimization model using the parameter estimator app of MATLAB/Simulink. Both approaches [38], [39] do not determine operational boundaries ④, parameters to characterize the input–output relationship ⑤, nor system state related parameters ⑥.

An MILP optimization model of an electrolyzer and a method for its partial parameterization from data are presented in [40]. System identification is applied to determine the parameters of a stack temperature dynamics model and to approximate the input–output relationship ⑤ through PLA with two segments with predefined boundaries. The parameters are derived from experimental data using least squares regression. Operational boundaries are also derived ④. However, certain parameters, such as system state ⑥ or storage related parameters ⑦, are not extracted [40].

An MILP optimization model for the waste heat recovery during steel production is presented by Kasper et al. [8]. Therein, most of the parameters (④, ⑥, and ⑦) are chosen by the authors and therefore not derived from data. However, the framework includes an approach to linearize nonlinear operational characteristics by means of PLA ⑤ [8].

The work of Li and Hong [41] aims at modeling energy flexible buildings. Analogous to the work in [40], parameters for existing multiperiodic *RC*-equivalence models ⑦ are determined through least squares regression from measured data. The resulting parameterized, nonlinear optimization model is then applied for model predictive control [41]. This approach does not allow for the derivation of parameters of operational boundaries ④, the input–output relationship ⑤, or system states ⑥ [41].

In the work of Peesel et al. [42], linear regression is applied to fit data of cooling system operation to the "Gordon–Ng universal model," a predefined model for the representation of the input–output relationship ⑤ of refrigeration machines, also suitable for MILP optimization. Operational boundaries ④ are also extracted, however system state ⑥ and storage related parameters ⑦ are not [42].

There are also a number of related works that apply machine learning approaches to the identification of parameters characterizing resources [16], [43], [44].

The flexibility of a lighting system and a heating and air conditioning system within a building are forecast by Barbero et al. [16]. Therein, parameters are fitted to an existing *RC*-equivalence model ⑦ using machine learning [16]. However, the objective of the work in [16] is the data-driven identification of flexibility potential and the forecasting of available flexibility at a given time step, not the derivation of parameters for optimization models ④–⑥.

Clustering is applied by Liisberg et al. [43], where a hidden Markov model (HMM) is used to identify occupant behavior of buildings based on time series data of smart meters. The occupant behavior is classified into three discrete states ⑥: "absent or asleep," "home, medium consumption," and "home, high consumption" [43]. Other parameters, such as holding durations, follower states, and ramp limits, are not extracted. The derivation of parameters for the fulfillment of ④, ⑤, and ⑦ is out of scope in the work of Liisberg et al. [43].

The method of Westermann et al. [44] uses logged binary control signals of actuators that are extracted from a programmable logic controller of the resource to identify system states ⑥ by applying the OTALA algorithm. The method also extracts a "transition time" of each state, which is similar to a holding duration. However, only one value per state is extracted, which is then used for timing anomaly detection [44].

TABLE 3. Fulfillment of Requirements for Parameter Derivation

Requirement	① - ③	④	⑤	⑥	⑦	⑧ - ⑨
Gade *et al.* [38]	■	■	■	■	●	■
Reinpold *et al.* [39]	■	■	■	■	●	■
Flamm *et al.* [40]	■	●	●	○	■	■
Kasper *et al.* [8]	■	○	●	○	○	■
Li *et al.* [41]	■	○	○	○	◐	■
Peesel *et al.* [42]	■	●	●	○	○	■
Barbero *et al.* [16]	Tab. 2	■	■	■	◐	■
Liisberg *et al.* [43]	■	■	■	◐	■	■
Westermann *et al.* [44]	■	■	■	◐	■	■

Fulfillment: ● – full ◐ – partial ○ – none ■ – n/a

Similar to the work in [43], other parameters (④, ⑤, and ⑦) are not part of the analysis.

The analysis of related work in deriving numerical parameters, summarized in Table 3, shows that approaches often require predefined, resource-specific models [8], [16], [38], [39], [40], [41], [42], such as *RC*-equivalent model formulations, and focus on only a subset of the parameters necessary. Machine learning is applied in determining the potential of energy flexibility or the forecast of a variable, but not to deriving a set of parameters for (optimization) models [16], [45]. The identification of system states and the derivation of related parameters is rarely focused, and if so requires specific source data, such as binary signals from actuators [44], or only determines the system states themselves without related parameters [43]. Thus, none of the existing approaches is able to be applied for the derivation of optimization model parameters ④–⑦ without further research or adaption. Furthermore, none of the works analyzed in this section describes a fully automated method for the derivation of all parameters necessary for the optimization model.

D. DEPENDENCY EXTRACTION FROM INFORMATION MODELS

This section describes related work in information modeling with a focus on the representation of dependencies and the subsequent extraction of dependency-related information as well as the fulfillment of requirements ⑧ and ⑨.

The FPD [21] features a process-oriented graphical modeling concept that allows for the modeling of processes, their connected resources, states, namely, energy, product, and information (see Section IV-D1). Flows connect processes and states [21]. A corresponding information model is standardized as in [46]. The works of the authors in [19] and [47] use AutomationML for the machine readable representation of an FPD information model. Nabizada et al. [48] introduced a graphical modeling tool for the FPD to decrease the modeling effort as well as a corresponding serialization format to make the FPD information model machine readable.

Jäger et al. [47] described an approach of utilizing information from an FPD information model to systematically match resources to processes. Dependencies, as defined in Section I, are not extracted, however connections between resources are part of their analysis ⑧, but types ⑨ are not.

Within the "interdependency model" of Hoang and Fay [19], relationships of parameters are described in order to find opportunities for reconfiguration. Previous work of Hoang et al. [49] includes a differentiation in correlative and restrictive dependencies ⑨. Thus, dependencies of resources ⑧ themselves are not extracted.

In the work of Jeleniewski et al. [17], relations of process-related parameters are extracted from an FPD information model, serialized by applying the tool of Nabizada et al. [48], and parameters themselves as well as parameter relations in the form of mathematical equations are represented in a semantic model. Process-related resources are part of the extraction ⑧, but relationship types ⑨ are not [17].

In addition to approaches directly utilizing the FPD for the representation of a system, a number of works [18], [23], [50] present approaches for the extraction of connectivity information from other standardized formats.

The works of the authors in [18] and [23] each present a method for the extraction of connectivity information from an AutomationML information model. Yim et al. [18] extracted connections modeled as "InternalLinks." The set of all "internal links" represents the connections of individual resources within the system [18]. Thambirajah et al. [23] presented a method to integrate the connectivity information into an existing application and present the result as a connectivity matrix. The goal is to simplify fault analysis [23]. Both approaches extract connectivity information from a standardized information model ⑧. However, only the correlative dependency is assumed ⑨.

The work of Barth and Fay [50] presents a method for the automatic generation of simulation models. In this method, resources representing parts of the system to be simulated, such as a pump, and their connections are extracted from a computer aided engineering exchange-based information model ⑧ and used to instantiate simulation objects of the resources in *Modelica* [50]. Types of the connections of resources ⑨, i.e., the type of each dependency, are not extracted.

Table 4 gives the summary of the analysis of related work in extracting dependencies from standardized information models. It becomes evident that whereas there are different approaches to extracting connectivity information and using it in other applications [17], [18], [23], [47], [50], few differentiate by the type of the connection between resources [18], [19], [23], [49], as required by requirement ⑨. None of the related work analyzed provides full fulfillment of ⑧ and ⑨.

E. RESEARCH GAP

The previous Sections III-A–III-D describe the analysis of related work in data models for flexible energy resources, preprocessing of time series data, deriving parameters for optimization models from time series data, and extracting

TABLE 4. Fulfillment of Requirements for Dependency Extraction

Requirement	① - ⑦	⑧	⑨
Jäger et al. [47]	■	◐	○
Hoang et al. [19, 49]	■	○	●
Jelenewski et al. [17]	■	●	○
Yim et al. [18]	■	●	◐
Thambirajah et al. [23]	■	●	◐
Barth *et al.* [50]	■	●	○

Fulfillment: ● – full ◐ – partial ○ – none ■ – n/a

dependencies from standardized information models. The fulfillment of the requirements is reported in a table for each research area (Tables 1–4). The tables summarize that none of the analyzed works fully fulfills the requirements of the respective research area by providing an integrated method for the derivation of parameters or extraction of dependency-related information. Furthermore, no work exists that covers all research areas. This underlines the novelty of the research contributions of this work.

IV. METHODOLOGY

This section presents a data model for parameters of flexible energy resources (Section IV-A) as well as algorithms for the data preprocessing (Section IV-B), the derivation of numerical parameters for individual energy resources (Section IV-C), and the extraction of dependency-related information (Section IV-D). The derivation of parameters for a system of flexible energy resources is described in Section IV-E. The implementation of the methodology and its algorithms is described in Section IV-F.

A. DATA MODEL

The fulfillment of requirement ① necessitates the development and utilization of a data model.

As described in Section II, the optimization model presented by Wagner et al. [7] contains a list of necessary parameters to describe the behavior of the resource to be optimized [7]. This includes resource-specific parameters, such as operational boundaries, parameters for the representation of the input–output relationship, and system state related parameters, such as boundaries by state, successor states, holding durations, and ramp limits [7]. Furthermore, parameters for storage systems, such as conversion efficiencies, are necessary [7]. As shown in Fig. 1, this parameter set must be extended to reflect system specific parameters, such as operational boundaries of the system as well as dependencies of individual energy resources within the system. The concept of dependencies of flexible energy resources, including the differentiation into correlative and restrictive, has been initially proposed by Schott et al. [10].

These parameters are to be persisted in a `.json` file to be made accessible for the parameterization of the optimization

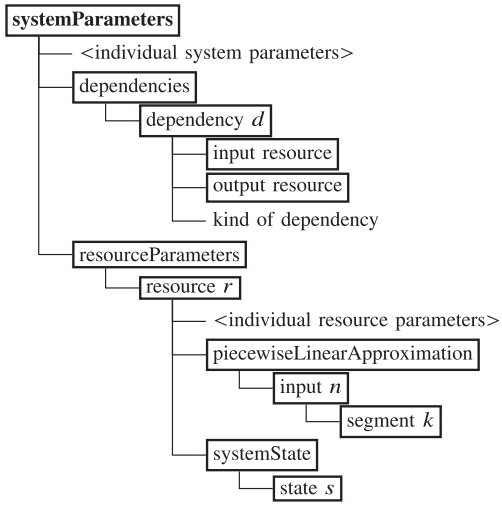


FIGURE 2. Structure of the data model (▭: nested element).

model. Fig. 2 shows the structure of the data model. Nested elements are created for all dependencies and resources. Analogous to the metamodel (see Fig. 1), each resource is described by a parameter set that consists of individual elements for parameters and nested elements for PLA and system states each.

To prevent infeasibility of the optimization model, default values are set for parameters in the absence of actual values. Three default values are required: all loss related parameters, lower operating boundaries, and lower ramp limits are set to 0 by default, and efficiencies or conversion factors are set to 1. Upper operating boundaries and upper ramp limits are ∞ (for the implementation, ∞ is approximated by `Double.MAX_VALUE`). The type of dependency relationships is set to "correlative" by default.

An object for system parameters is created and all parameters are added during the extraction (see Section IV-E). Eventually, this object is then persisted to the data model.

B. DATA REQUIREMENTS AND PREPROCESSING

The time series data must contain a sufficiently long interval of resource operation to extract all possible behaviors [14]. To extract meaningful parameters from the data, an appropriate temporal resolution of the data is required. An analysis of the temporal resolution of time series data for parameter derivation shows that a temporal resolution of seconds and minutes represents the resource behavior sufficiently well, whereas an hourly temporal resolution averages out certain effects, such as load changes [14]. Applying the findings of Hechelmann [14] to this work means that the temporal resolution of the data should be at least in the range of minutes.

The time series dataset must include various time series data columns of measurements, as depicted in Fig. 3. Measurements for the inputs of the system and its output as well as for each resource are required. In case of storage systems, time series data columns for the actual flow inside the storage

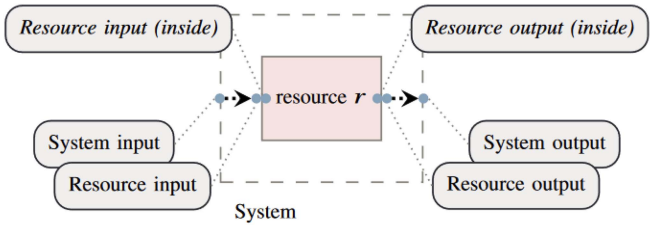


FIGURE 3. Necessary measurements.

Algorithm 1: Preprocessing of Time Series Data.

Data: $\mathcal{X}$, $\mathcal{Y}$
Result: $\mathcal{X}_p$, $\mathcal{Y}_p$
Function `doPreProcessing(`$\mathcal{X}$, $\mathcal{Y}$`)`:
  $R^2 \leftarrow$ calculateR2($\mathcal{X}$, $\mathcal{Y}$)
  pValue $\leftarrow$ calculatePValue($\mathcal{X}$, $\mathcal{Y}$)
  stationary $\leftarrow$ checkForStationarity($\mathcal{X}$, $\mathcal{Y}$)
  if $R^2 < \varrho$ && *pValue* $\geq \alpha$ && *stationary*==*false* **then**
    return null `// no correlation`
  else
    $\mathcal{X}_p \leftarrow$ identifyAndReplaceOutliers($\mathcal{X}$)
    $\mathcal{Y}_p \leftarrow$ identifyAndReplaceOutliers($\mathcal{Y}$)
    return $\mathcal{X}_p$, $\mathcal{Y}_p$

["resource input (inside)" and "resource output (inside)"], after conversion-induced losses (efficiency) occurred, are also necessary. These time series data columns provide insight into the actual "usable" within the resources and is used to determine the efficiencies of the conversions ("storing" and "retrieving").

As described in Section III-B, data preprocessing must fulfill requirement ② to ensure that the data to be used actually represent correlations of either two or more time series data columns, or one time series data column and the corresponding time stamps, and requirement ③ to remove or replace inaccurate values, such as outliers, thereby increasing the accuracy of the parameters that are based on the data.

As described in Algorithm 1, statistical indicators are used to determine the significance of the relationships in the data, as well as the expected impact of one time series data column on another, both the same and different time series. Correlations between columns of time series data are determined by means of interpreting the coefficient of determination (R^2). The R^2 is a measure of how well a linear regression model of time series data columns approximates the measured values for the predicted time series data column(s). Values range from 0 to 1, with 1 being the best fit. If the R^2 is greater than a target value $\varrho = 0.9$, the prediction of the regression model is good enough [51]. In the case, of more than one time series data column to be used for the prediction, the adjusted R^2 is determined. The interpretation is analogous to the R^2.

As a second measure, the significance of the relationship of two time series data columns is determined by means of the p-value. It is common practice, to set a p-value of $\alpha = 0.05$ as a threshold for the significance of the relationship [13], [34], i.e., the relationship of any data with a p-value below 0.05 is deemed to be statistically significant.

If there is more than one time series data column representing the additional measurements, such as temperature, the relevant time series data columns must be determined: the R^2 and p-value of each combination of time series data columns and the output time series data column are determined and checked to see if the thresholds are met.

To ensure reliable results of the statistical analyses, a check for stationarity by means of applying the augmented Dickey–Fuller test is conducted. The augmented Dickey–Fuller test involves estimating a regression model with lagged differences, calculating a test statistic to compare against critical values, ultimately indicating whether the data are stationary or not [33].

The (adj.) R^2 and the p-value of the time series data are calculated and the check for stationarity is conducted, and only if the respective thresholds are met, further preprocessing is performed (see Algorithm 1). This ensures that only representative data are used for the parameter derivation.

If there is no statistically significant relationship between the different time series data columns, new data must be identified to derive parameters.

Inaccurate values, such as outliers, are identified using percentiles, as suggested by the authors in [14] and [15]. In this work, the 5th and 95th percentiles are used [14], [15]. The identified values are then replaced by means of linear interpolation with the pre and succeeding value [35]. In case the first of last values identified as erroneous, extrapolation is performed. Any preprocessed time series data column is labeled with an additional index p.

C. DERIVATION OF NUMERICAL PARAMETERS

This section describes the method of deriving numerical parameters from preprocessed time series data for the fulfillment of requirements ④–⑦. This method must be applied once per resource. The compilation of parameters of all resources within a system is described in Section IV-E.

The set of resource parameters can be divided into subsets.

1) **P1** upper and lower boundaries of decision variables, such as minimum power input of a resource.
2) **P2** parameters for the characterization of the input–output relationship, such as an efficiency of the conversion or parameters for a PLA.
3) **P3** system states and related parameters, such as state sequences, holding durations, and ramp.
4) **P4** storage system related parameters, such as efficiencies, losses of the storage, and degradation.

In the following, a consistent notation is applied: an energy resource r has $|\mathcal{N}|$ input flows $n \in \mathcal{N}$. The input–output relationship can be characterized by means of $|\mathcal{K}|$ linear segments of PLA $k \in \mathcal{K}$ per input flow n. The characteristics of

a resource are further described by means of $|\mathcal{S}|$ system states $s \in \mathcal{S}$.

1) **P1** OPERATIONAL BOUNDARIES

Decision variables representing flows or storage systems' SOC are limited by the operational boundaries of the resource. Thus, lower *lb* and upper boundaries *ub* of each variable *var* are derived from the respective preprocessed time series data column $\mathcal{X}_{p,r}$ by means of extracting the minimum (1) and maximum value (2)

$$\text{lb}_{\text{var},r} = \min\left\{\mathcal{X}_{p,r}\right\} \tag{1}$$

$$\text{ub}_{\text{var},r} = \max\left\{\mathcal{X}_{p,r}\right\}. \tag{2}$$

2) **P2** INPUT–OUTPUT RELATIONSHIP

As the optimization model is realized as an MILP, the input–output relationship must be represented linearly. Thus, linear regression is performed. The analysis of related work in Section III-C supports this approach [40], [41], [42]. In case of a nonlinear relationship, is it approximated using PLA [8], [26], [40]. The analysis of existing generic optimization models for flexible energy resources also supports this approach [7].

Algorithm 2 describes the derivation of parameters of the input–output relationship, namely, slope and intercept of each linearized segment, as well as lower and upper bounds of each segment, if PLA is used. Algorithm 2 returns a list of lists of the object `PLA` $\mathcal{K}$ that stores parameters for all inputs $n \in \mathcal{N}$ and potentially all linear segments k. This structure corresponds to the nested elements of the PLA parameters in Fig. 2.

First, (multiple) linear, least squares regression with preprocessed time series data columns $\mathcal{X}_{p,r}$ (input) and $\mathcal{Y}_{p,r}$ (output) is performed. The input data $\mathcal{X}_{p,r}$ can consist of one or more columns, and which time series data columns are necessary is determined during data preprocessing (see Section IV-B). Whether a linearized representation of the relationship is sufficient or PLA is necessary is determined by means of comparing the linear regression's R^2 to a target value ς, e.g., $\varsigma = 0.9$, in case of multiple linear regression ($|\mathcal{N}| > 1$), the adjusted R^2 is used. If the (adjusted) R^2 of the (multiple) linear regression is greater than the target value ς, values for slope per input n of the (multiple) linear regression are used and returned in list of parameters $\mathcal{K}$. The intercept is only extracted once per relationship, as the relationship is defined as shown in (3). All other intercepts are set to 0

$$\text{output} = \left(\sum_{n\in\mathcal{N}} \text{slope}_n \cdot \text{input}_n\right) + \text{intercept}. \tag{3}$$

Otherwise, (multiple) linear segmented regression is performed to determine parameters of the PLA. For the segmented regression, initial starting values for the breakpoints must be set. Quantiles are used to approximate breakpoint locations. In decreasing number of quantiles (quantile distance: $\{0.05, 0.1, 0.2, 0.25, 0.5\}$), quantiles are set as breakpoints and the segmented regression is tried. If the segmented regression is infeasible, a lower number of breakpoints are tried until either a set of parameters is calculated or all possible values for the quantile distance are tried. In this case, the parameters for slope and intercept of the linear regression are used.

If the segmented regression does not show a significant improvement over the linear regression in terms of increased R^2 by an improvement value ψ (e.g., $\psi = 0.025$), the results of the linear regression are used. This is done since the PLA requires binary variables for its representation in the optimization model [7] and thus increases the computational complexity [26], [52]. Otherwise, the results are returned in the list of lists of parameters $\mathcal{K}$ per input n, which includes slope, intercept, as well as lower and upper bounds per segment k.

Algorithm 2: Parameters for Input–Output Relationship.

```
Data: X_p, Y_p            // preprocessed time series data
Result: List<List<PiecewiseLinearApproximation>> K
Function getParameterIORel(X_p, Y_p):
    linReg ← doLeastSquaresRegression(X_p, Y_p)
    if linReg.R² > ς then
        while n <= |N| do
            K[n][0].slope ← linReg[n].slope
            if n == 0 then
                K[n][0].intercept ← linReg[n].intercept
            else
                K[n][0].intercept ← 0
    else
        if successfulCalc == true then
            for dist in quantilDist do
                try:
                    segReg ← doSegRegression(X_p, Y_p, dist)
                    successfulCalc ← true
                catch Exception:
                    segReg ← null
        if segReg.R² > (linReg.R² + ψ) && segReg!=null then
            while n <= |N| do
                while k <= segReg.size do
                    K[n][k] ← segReg[n][k]
        else
            while n <= |N| do
                K[n][0].slope ← linReg[n].slope
                if n == 0 then
                    K[n][0].intercept ← linReg[n].intercept
                else
                    K[n][0].intercept ← 0
    return K
```

3) **P3** SYSTEM STATE RELATED PARAMETERS

System states are characterized by state boundaries, follower states, holding durations, and respective ramp limits (see Fig. 2) [7].

It is important to note that the term "state" in the context of *system states* is different from the term "state," which describes part of an FPD information model, as used in Section IV-D.

Recorded system states can be used as a starting point to extract system state related parameters. In case no system

Algorithm 3: Clustering of System States From Time Series Data.

Data: $\mathcal{X}_p$, $\mathcal{Y}_p$
Data: numberOfSysStates
Result: S_r // array of active system states
Function getSystemStates($\mathcal{X}_p$, $\mathcal{Y}_p$,*numberOfSysStates*):
  if *numberOfSysStates* == *null* **then**
    numberOfSysStates ← 3
  model ← PoissonHMM(numberOfSysStates)
  model.fit($\mathcal{X}_p$, $\mathcal{Y}_p$)
  S_r ← model.predict($\mathcal{X}_p$, $\mathcal{Y}_p$)
  return S_r

Algorithm 4: State Boundaries for State s.

Data: S_r // array of active system states
Data: $\mathcal{X}_p$, $\mathcal{Y}_p$ // relevant time series data columns
Result: SystemState $\mathcal{S}_s$
Function getStateBoundaries(s, S_r, $\mathcal{X}_p$, $\mathcal{Y}_p$):
  if $S_p[t]$ == s **then**
    $S_{p,s,\mathcal{X}}$[counter] ← $\mathcal{X}_p[t]$
    $S_{p,s,\mathcal{Y}}$[counter] ← $\mathcal{Y}_p[t]$
    counter++
  $\mathcal{S}_s$.minInput ← min($S_{p,s,\mathcal{X}}$)
  $\mathcal{S}_s$.maxInput ← max($S_{p,s,\mathcal{X}}$)
  $\mathcal{S}_s$.maxOutput ← max($S_{p,s,\mathcal{Y}}$)
  return $\mathcal{S}_s$

Algorithm 5: Possible Follower States for State.

Data: S_r // array of active system states
Result: $\mathcal{S}_{F,s}$ // list of follower states
Function getFollowerStates(s, S_r):
  while $t < S_p.length-1$ **do**
    if $S_p[t] == s$ && $S_p[t]! = S_p[t+1]$ && $\mathcal{S}_{F,s}.contains(S_p[t+1])$ == *false* **then**
      $\mathcal{S}_{F,s}$.add($S_p[t+1]$)
  return $\mathcal{S}_{F,s}$

states have been recorded, a purely data-based method must be adopted, wherein system states are derived from time series data. This method is described in the following.

a) System states from time series data: "Clustering techniques apply when there is no class to be predicted but rather when the instances are to be divided into natural groups" [53]. As described by Liisberg et al. [43], the HMM is a promising approach to identify states within observed time series data.

In this work, an algorithm based on the *Poisson* HMM is chosen. This is suitable because the time series data of the resource operation can be seen as count data and thus represented by a Poisson distribution [43]. Furthermore, an underlying HMM consisting of $|\mathcal{S}|$ discrete states is assumed. However, it is unknown which hidden state s is active at which time step t.

To identify the most likely active state at each time step, the model is trained on the time series data. This involves estimating parameters for the initial state probabilities, state transition probabilities, and the parameters of the Poisson distributions in each state. Once the model parameters are estimated, the HMM can be used to infer the most likely sequence of hidden states given the observed time series. This inference is performed using the "Viterbi" algorithm [54].

The algorithm for generating a time series data column of active system states, described in Algorithm 3, requires the a priori specification of a number of states to be identified. The active states are then inferred based on the time series data columns of the input and output of the resource. Either the number of system states of the resource is known and can be used for generation, or a number must be assumed.

An analysis of the impact of the number of system states on the alignment of optimized schedule and the resource behavior shows that three system states are often well suited to represent the resource while not overly increasing the computational complexity of intraday scheduling [26]. Thus, a number of three system states is chosen as the default number of system states in the absence of expert knowledge of the resource.

The following paragraphs describe the extraction of parameters for state boundaries, state sequences, holding durations, and ramp limits. The respective algorithms need to be applied once per system state.

b) State boundaries: System states are characterized for optimization by bounds on the continuous decision variables of the resource representing the input and output flow or the SOC (in the case of storage systems). [7]

As described in Algorithm 4, the time series data columns of the input $\mathcal{X}_{p,r}$ and the output $\mathcal{Y}_{p,r}$ of the resource are segmented according to the active system state s. For storage resources, this is done for the SOC instead of the input and output flows. Then, the minimum and maximum value of the input flow and the maximum output flow of state s is derived.

c) Follower states: Algorithm 5 describes how state sequences can be derived from time series data of active system states. Therein, all succeeding states of state s are extracted by iterating over the time series data of active system states and compiling a unique set of all follower states $\mathcal{S}_{F,s}$.

d) Holding durations: Analogous to the determination of state sequences, minimum and maximum holding durations per state can be derived from the time series data of active system states. Algorithm 6 describes the extraction: for all occurrences of state s within the time series data of active system states, all succeeding occurrences of this state are counted and the corresponding time steps are used to determine the active length of one occurrence. The minimum and maximum length are then calculated and used to set the respective parameters for the minimum and maximum holding duration of state s. If the maximum duration is greater than a threshold value, e.g., 8 h, the maximum holding duration is set to ∞.

e) Ramp limits: Ramp limits are lower and upper limits of the rate of change of a decision variable $\frac{P}{\mathrm{d}t}$, discretized for the optimization as $\frac{\Delta P}{\Delta t}$ [7]. The unit of the ramp limits is power per time. Respective ramp limits restrict the input and output flows of a resource. The optimization model requires

Algorithm 6: Min./Max. Holding Duration for State.

```
Data: S_r                 // array of active system states
Data: timeStamp           // array of time stamps of S_r
Result: SystemState S_s
Function getHoldingDurations(s, S_r, timeStamp):
    List<Double> dur // list of all dur. of state s
    while t < S_r.length do
        if S_r[t] == s then
            counter++
        else
            dur.add(timeStamp[t] - timeStamp[t-counter])
            counter ← 0
    S_s.minDuration(min(dur))
    if max(dur)>threshold then
        S_s.maxDuration(∞)
    else
        S_s.maxDuration(max(dur))
    return S_s
```

Algorithm 7: Slopes of Flow of Segment of Active State.

```
Data: tsd, timeStamps     // segment with active state s
Result: slopes
Function getSlopes(tsd,timeStamps):
    segCount ← 0
    breakpoints ←
    while segCount < breakpoints.length do
        if segCount == 0 then
            tsdSubList ← tsd.subList(0, breakpoints[segCount])
            timeStampSubList ← timeStamps.subList(0,
              breakpoints[segCount])
        else
            tsdSubList ← tsd.subList(breakpoints[segCount-1],
              breakpoints[segCount])
            timeStampSubList ←
              timeStamps.subList(breakpoints[segCount-1],
              breakpoints[segCount])
        slopeSegment ← slope(tsdSubList[0], tsdSubList[last],
          timeStampSubList[0], timeStampSubList[last])
        slopes.add(abs(slopeSegment))
        segCount++
    return slopes
```

parameters for ramp limits per state [7]. Thus, Algorithms 7 and 8 are applied once per state. The time series data columns are divided into segments *tsd* where the respective state is active. For those segments, values for slope of the input flow are extracted as described in Algorithm 7.

Time stamps where the sign of the slope of the flow changes, hereafter called "breakpoints," must be identified in order to extract individual values of the slope. A threshold for a minimum absolute difference between two consecutive time steps is defined as 10% of the maximum value of the time series data column. Only when this threshold is exceeded, a new breakpoint is identified. This "ignores" small changes in values, such as measurement errors or small fluctuations, and reduces the number of breakpoints.

As described in Algorithm 7, ramp parameters are then determined by extracting the absolute value of the slopes of the time series data and their corresponding time steps between two breakpoints and added to a list of slopes.

Algorithm 8: Ramp Limits for State s.

```
Data: X_P, S_r, timeStamps
Result: SystemState S
Function getRampLimits(s, S_r, X_P, timeStamp):
    X[][] ← segmentTSDByActiveState(X_P, S_r, s)
    T[][] ← segmentTSDByActiveState(timeStamps, S_r, s)
    counter ← 0
    while counter < T.length do
        rampInput.add(getSlopes(X[counter], T[counter]))
    S.minRampInput ← min(rampInput)
    S.maxRampInput ← max(rampInput)
    return S
```

As described in Algorithm 8, all slope values of each segment, determined by means of Algorithm 7, are then added to a list of slopes. The minimum and maximum ramp parameters per state are then determined from the full list by calculating the minimum and maximum values, respectively.

4) **P4** STORAGE SYSTEMS

A storage system is characterized by its capacity, represented by lower and upper bounds of the SOC, and conversion efficiencies [7].

a) Capacity boundaries: Analogous to (1) and (2), the minimum and maximum storage capacities can be determined, respectively.

b) Efficiencies: The respective efficiencies of the storage input and output are defined as shown in (4) and (5), respectively. The parameters are the average value of the individually calculated efficiencies [see (6)]

$$\eta_{\text{input},t} = \frac{P_{\text{input side, inside storage},t}}{P_{\text{storage input},t}} \tag{4}$$

$$\eta_{\text{output},t} = \frac{P_{\text{storage output},t}}{P_{\text{output side, inside storage},t}} \tag{5}$$

$$\bar{\eta}_{\text{in-/output}} = \frac{1}{|\mathcal{T}|} \cdot \sum_{t \in \mathcal{T}} \eta_{\text{in-/output},t}\,. \tag{6}$$

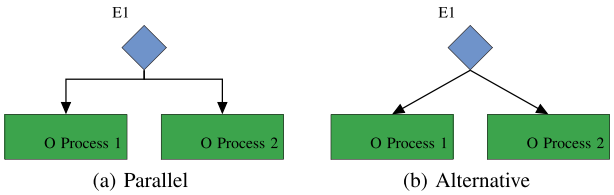
D. EXTRACTION OF DEPENDENCY-RELATED INFORMATION

This section describes the use of the FPD for the representation of the structure of the system and its serialization (Section IV-D1). Furthermore, the method for the extraction of dependencies and their types is outlined in Section IV-D2.

As an alternative to extracting dependency-related information from an information model of the system structure, e.g., in case, that such an information model does not exist, the dependencies can be added to the data model (see Fig. 2) manually.

1) INFORMATION MODELING TO REPRESENT OF THE SYSTEM

The modeling of the structure of the system is realized by means of the graphical modeling concept of the FPD [21], [46] and the corresponding tool developed by Nabizada et al. [48].



(a) Parallel (b) Alternative

FIGURE 4. (a) Parallel and (b) alternative flows [21].

As underlined by the use cases of [17], [19], and [47], the FPD is well suited for the representation of connectivity information of parts of systems, e.g., dependencies of flexible energy resources within a system. Furthermore, practitioners can easily construct an information model of the system to be optimized thanks to the graphical modeling interface [48].

The graphical modeling of an FPD information model, as shown in Fig. 5, visualizes which states, namely, products, energies, or information, are used by a process, which is carried out within a resource. For the nomenclature of the graphical modeling, refer to Fig. 5(a). All processes and resources are located within a system boundary (dashed rectangle). An output flow of a process can serve as an input flow of another process by means of an intermediary state. The standard explicitly states the location of identifying text of different symbols [21].

Different types of flows between states and processes can be used to represent the type of connection. The flows are "parallel" [see Fig. 4(a)] and "alternative" [see Fig. 4(b)] connections, which are introduced by the standard [21]. The same flow types can be used to connect processes to states [21]. These flow types can be used to represent the type of dependency, i.e., correlative (parallel) and restrictive (alternative), as explained in Section II.

Fig. 5(a) shows an exemplary information model of a system with one energy input and one energy output and two processes with one energy input each. The output of "Process 1" is energy E2, which is also the input of "Process 2." This example system contains two processes with corresponding resources "Resource 1" and "Resource 2." The resources have a correlative dependency. Fig. 5(c) depicts an FPD information model with a restrictive dependency of the resources by means of "alternative flows."

2) DEPENDENCIES OF RESOURCES

This section describes the algorithm for the extraction of dependencies of resources within a system of flexible energy resources for the fulfillment of requirements ⑧ and ⑨.

As shown in Fig. 2, a dependency relationship is characterized by two lists of resources involved in a dependency relationship. One list contains all connected outputs and another list contains all connected inputs. Furthermore, the type of dependency (either correlative or restrictive) is also part of the representation.

Algorithm 9: Extraction of Dependencies.

```
Data: fpd                                   // serialized FPD model
Result: dependencies                        // see Fig. 2
Class dependencyExtraction():
    listOfProcesses ← getProcesses(fpd)
    listOfProcesses ← enrichListByStates(fpd)
    listOfProcesses ← enrichListByFlowType(fpd)
    dependencies ← getDependencies(listOfProcesses)
Function getDependencies(listOfProcesses):
    dependencies.add(getDepResources(listOfProcesses))
    dependencies.add(getDepSystemInput(listOfProcesses))
    dependencies.add(getDepSystemOutput(listOfProcesses))
    return dependencies
```

Algorithm 9 describes the method of the extraction. Therein, the serialized FPD information model of the system of flexible energy resources *fpd* is parsed. A list of processes *listOfProcesses* as well as the connected states are extracted.

Algorithm 9 contains two functions for the extraction of dependencies: one for the extraction of dependencies of resources (Algorithm 10) and one for determining dependencies of the "first" or "last" resource and the system input or output (Algorithm 11).

In Algorithm 10, processes are mapped to associated processes by analyzing connecting flows (see Fig. 4) and intermediate states based on the unique identifiers of each component of the information model. This is done for all processes. Resources are then mapped to processes since resources are assigned to processes during the information modeling. The type of dependency is distinguished by checking the type of flow of the connection, as described in Section IV-D1.

Algorithm 11 extracts dependencies of the system input and the respective "first" process(es) of the system. For this purpose, the algorithm iterates over the list of states *listOfStates*. If the incoming port is empty, a system input-relevant state has been identified. Then, all processes connected to this state must be identified and saved to the list of dependencies. Analogous to Algorithm 10, the type of dependency is identified. Dependencies of the system output are identified in a similar way.

E. SYSTEM SPECIFIC PARAMETERS

This section describes the derivation of operational boundaries of the system (Section IV-E1). Furthermore, a method for determining the most appropriate temporal resolution of the optimization model is presented (Section IV-E2). Finally, the approach to compile an instance of the data model presented in Section IV-A to persist all determined parameters and dependency-related information is described in Section IV-E3.

1) OPERATIONAL BOUNDARIES OF THE SYSTEM

Analogous to the derivation of the operational boundaries of individual resources, the operational boundaries of the system, such as the limits of the system's grid connection, must also be derived from the corresponding time series data columns. This can be done by applying the algorithm described in

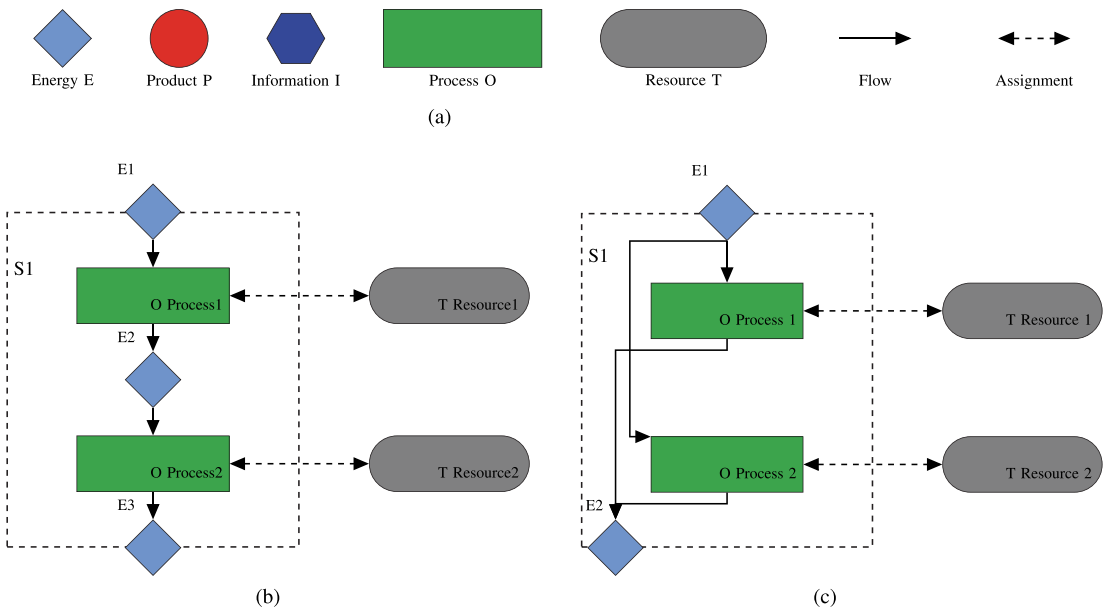


FIGURE 5. FPD information model of exemplary system of flexible energy resources. (a) Symbols of the FPD [21]. (b) Correlative dependency. (c) Restrictive dependency.

Algorithm 10: Extraction of Dependencies Between Resources.

```
Function getDepResources(listOfProcesses):
    for processA in listOfProcesses do
        dependency.output(resources(processA))
        for processB in listOfProcesses do
            alignment ← compareStates(processA, processB)
            if alignment == true then
                dependency.input(resources(processB))
                if flowType(processA.stateOutput)
                  =="Alternative" &&
                  flowType(processB.stateInput) ==
                  "Alternative" then
                    dependency.type ← "restrictive"
        if dependency.input.size>0 &&
          dependency.output.size>0 then
            dependencies.add(dependency)
    return dependencies
```

Algorithm 11: Extraction of Dependencies (System Input).

```
Function getDepSystemInput(listOfProcesses,listOfStates):
    for state in listOfStates do
        if state.incoming.isEmpty then
            dependency.output("SystemInput")
            for process in listOfProcesses do
                if process.inputStates==state then
                    dependency.input(resources(process))
            if dependency.input.states.flowType ==
              "Alternative" then
                dependency.type ← "restrictive"
            dependencies.add(dependency)
    return dependencies
```

Section IV-C1 to time series data columns representing the systems' input and output flows.

2) TEMPORAL RESOLUTION OF THE OPTIMIZATION MODEL

For the optimization, it is necessary to specify a minimum and maximum holding duration per state, expressed in number of time steps [7]. Therefore, the largest possible temporal resolution must also be determined, since the length of each holding duration must be converted from time to the number of optimization time steps. This must be done so as not to increase the computational complexity unnecessarily [52].

An analysis of the required temporal resolution shows that a temporal resolution of 15 min is still sufficient for planning the resource operation [52]. The length of one or more time steps must also match the trading interval of the energy market, i.e., the 15-min interval of the intraday market. However, smaller resolutions may be appropriate to match holding durations. In general, the computational complexity of MILP optimization models increases with the number of binary variables (such as those necessary for PLA or system states) [26]. As the binary variables are introduced for each time step of the optimization, the number multiplies by the number of time steps into which the optimization horizon is divided. Thus, the computational complexity is also directly dependent on the temporal resolution.

Due to numerical inaccuracies of the solver that occur when rounding fractions to 16-digit decimals [55], the set of possible temporal resolutions *possTempRes* contains resolutions of 1.5, 3, 3.75, 7.5, and 15 min. These resolutions can be

Algorithm 12: Selection of Temporal Resolution.

```
Data: allHoldDur
Result: tempRes
Function getTempRes(allHoldingDurations):
    minDist ← ∞
    for oneTempRes in possTempRes do
        avgDist ← getAvgDist(oneTempRes, allHoldDur)
        if avgDist<minDist then
            minDist ← avgDist
            tempRes ← oneTempRes
    return tempRes
Function getAvgDist(oneTempRes, allHoldDur):
    for oneHoldDur in allHoldDur do
        sumDist ← sumDist + onePossTempRes%oneHoldDur +
          oneHoldDur%oneTempRes
    avgDist ← (sumDist/(2·allHoldDur.length))/oneTempRes
    return avgDist
```

expressed as decimals with a small number of valid digits: 0.025, 0.05, 0.0625, 0.125, and 0.25 h.

The algorithm for selecting the appropriate temporal resolution, described in Algorithm 12, is based on the Greedy algorithm. It determines which element of the set of possible temporal resolutions fits all the holding durations *allHoldDur*. The fit is evaluated by calculating the average distance between each holding duration and each element of the set of possible temporal resolutions, taking into account the remainder of the division of the two temporal resolutions. The average distance is weighted by the length of the possible temporal resolution to favor longer resolutions to reduce computational complexity [52].

3) COMPILATION OF PARAMETERS

This section shows how the methodology described in this section can be applied for the derivation of parameters.

First, the specification of time series data columns corresponding to the respective measurement points, as shown in Fig. 3, is necessary. Then, as shown in Algorithm 13, data preprocessing (see Section IV-B) must be conducted, to ensure meaningful parameter derivation. Following, as shown in, the algorithms described in Section IV-C are applied. This involves deriving operational boundaries, parameters of the input–output relationship, or storage related parameters, as well as determining parameters for the representation of system states. Algorithm 13 is applied once per resource.

The flowchart depicted in Fig. 6 further visualizes the application of the algorithms described in the previous sections.

The operational boundaries of the system are also derived from preprocessed time series data ("get system parameters"), as described in Section IV-E1.

Dependencies of resources within the system are extracted from an FPD information model, as described in Section IV-D2. The alternative approach to applying Algorithm 9, i.e., manually adding dependencies, is depicted by means of the dashed line.

Finally, all parameters and dependencies are persisted to the data model (see Section IV-A).

Algorithm 13: Compilation of Resource Parameters.

```
Data: tsd, numberOfStates (optional)
Result: ResourceParameters resPara
Function getResourceParameters(tsd, numberOfStates):
    tsd_p ← doProProcessing(tsd) // Algorithm 1
    if tsd_p != null then
        resPara.operationalBoundaries ← {Eq. 1 & 2}(tsd_p)
        if tsd.storageTsd == null then
            resPara.piecewiseLinearApproximation ←
                getParameterIORel(tsd_p) // Algorithm 2
        else
            resPara.storageParameters ← {Eq. 4 – 6}(tsd_p)
        if numberOfStates==null then
            numberOfStates=3
        if numberOfStates>0 then
            SystemStates sys
            S_r ← getSystemStages(tsd_p, numberOfStates)
                // Algorithm 3
            sys.addUniqueStates(S_r)
            for state in sys do
                state.boundaries ← getStateBoundaries(state,
                    S_r, tsd_p) // Algorithm 4
                state.followerStates ← getFollowerStates(state,
                    S_r) // Algorithm 5
                state.holdingDurations ←
                    getHoldingDurations(state, S_r, tsd_p)
                    // Algorithm 6
                state.rampLimits ← getRampLimits(state, S_r,
                    tsd_p) // Algorithm 8
                sys.add(state)
            resPara.systemStates ← sys
    return resPara
```

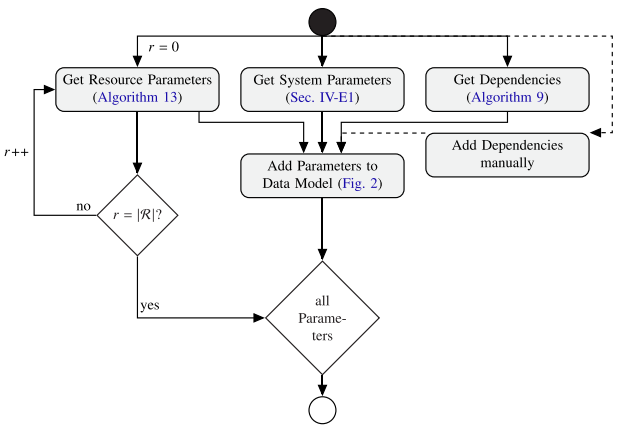


FIGURE 6. Flowchart of the methodology.

F. IMPLEMENTATION OF THE METHODOLOGY

The methodology and corresponding algorithms described in Sections IV-A–IV-E are implemented in Java.[1] The clustering algorithm described in Algorithm 3 is implemented in Python.

To determine PLA-related parameters, the package *RCaller* [56] is used to access the statistical analysis capabilities of *R*, including the library *segmented* [57] from within Java. The information model of the system is built using the

[1]The data model and implementation are available at https://github.com/lukas-wagner/MethodologyParametersForOptimizationModels.

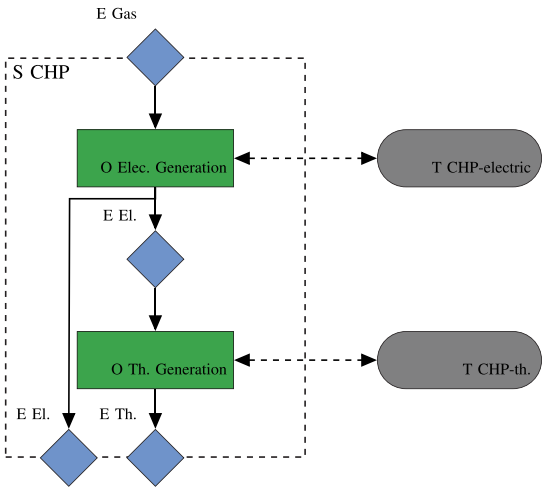


FIGURE 7. FPD information model of the CHP system.

web-based tool for the FPD,[2] which allows the generation of a serialized information model from the graphical modeling [48].

Classes (SystemParameters, ResourceParameters, PLA, SystemStates, and Dependencies) are used as objects to save extracted parameters. These objects are persisted to a `.json`-file of the data model (see Section IV-A) using gson [58].

V. VALIDATION OF THE METHODOLOGY

This section describes the application and validation of the methodology described in Section IV for the parameterization of an optimization model of a combined heat and power (CHP) system. The input data for the algorithms for the derivation of parameters (see Section IV) are time series data of the resource operation as well as an FPD information model representing the structure of the system. Applying the algorithms results in an instance of the data model (Section IV-A/Fig. 2).

The data model can then be used to parameterize an optimization model, using design patterns presented by Wagner et al. [7]. This is described in Section V-D.

Section V-E provides a comparison of the methodology and its results.

The CHP system consists of a gas-fired generator (gas-fired gen.), which generates electric power "E El." from gas "E Gas" (resource 1). Furthermore, thermal energy "E Th." is produced by means of a heat exchanger (heat ex., resource 2) from the exhaust gas of the gas-fired generator, which is proportional to the output of the gas-fired generator. The structure of the system is shown in Fig. 7. The serialized version of this information model is used for the extraction of dependencies by means of Algorithm 9.

The algorithms described in Section IV were executed as shown in Fig. 6 on Windows 11 with an AMD Ryzen 7 PRO 5850 U processor and 32 GB RAM. The total calculation time was 5.4 s. As the parameterization of the optimization model is not necessary on a recurring basis, especially not during the solving of the optimization model, this time is negligible.

The methodology has also been applied to derive parameters for an optimization model of a system of electrolyzers in [59].

A. DERIVATION OF PARAMETERS

This section describes the results of the preprocessing of the time series data, as described in Section IV-B. The preprocessed time series data columns are then used for the derivation of parameters in the following, applying the algorithms described in Section IV-C.

Fig. 8 shows a comparison of the unpreprocessed time series data columns and the processed time series data columns by means of a boxplot. Therein, it becomes visible that inaccurate values, such as outliers, have been replaced, but the dataset itself has not been altered excessively. The size of the time series dataset has not been reduced during preprocessing ($|\mathcal{T}| = 30\,000$). This size of the dataset is sufficiently large, as all possible resource behaviors are captured.

Table 5 gives the parameters extracted by application of the algorithms described in Section IV-C. Therein, operational boundaries and parameters characterizing the input–output relationship of both resources are listed. In Fig. 9(a), it is apparent that a linearized representation of the input–output relationship of both resources is sufficient [R^2 (generator) = 0.997]. The correlation for the heat exchanger is weaker, but still above the defined threshold [R^2 (heat ex.) = 0.9].

Fig. 10 shows the visual alignment of the time series data column of "input" with the detected time series data column of active system states (with Algorithm 3). Therein, it is shown that one state (red shading, i.e.,"operation") occurs when the resource is in operation and another state when the resource is not in operation (gray shading, i.e., "off").

Table 6 contains all system state related parameters of resource 1, extracted with the algorithms described in Section IV-C3. These parameters coincide with Fig. 10 (red: state 0, gray: state 1). Resource 2 (heat exchanger) does not exhibit any system states.

The limits for the flows of each state can be interpreted as either the input of the gas-fired generator is between 0 and 1452 kW (state 1) or in between 0 and 2912 kW (state 0). This then affects the maximum output flow to 536 kW (state 1) or 1101 kW (state 0). This can also be seen in Fig. 9(a) (red), where most data points of the resource operation are above output/$\text{output}_{\text{max}}$ of 0.5. The parameter for the maximum output flow in state 0 coincides with the upper operational boundary of the resource (see Table 5).

State 1 has a minimum holding duration of 0.5 h, i.e., after a transition from state 0 to state 1, the gas-fired generator must remain in state 1 for 0.5 h. For state 0, this minimum holding duration is 8.15 h. This means that once state 0 has been entered, a transition to another state cannot occur until the time of 8.15 h has passed. In this case, this means that once the gas-fired generator has entered state 0 (operation), it

[2][Online]. Available: https://demo.fpbjs.net

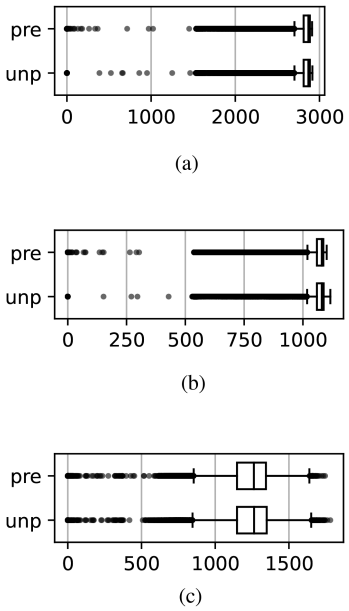


FIGURE 8. Boxplot of time series data, before (unp.) and after preprocessing (pre.). (a) Gas-fired generator (input). (b) Gas-fired generator (output). (c) Heat exchanger (output).

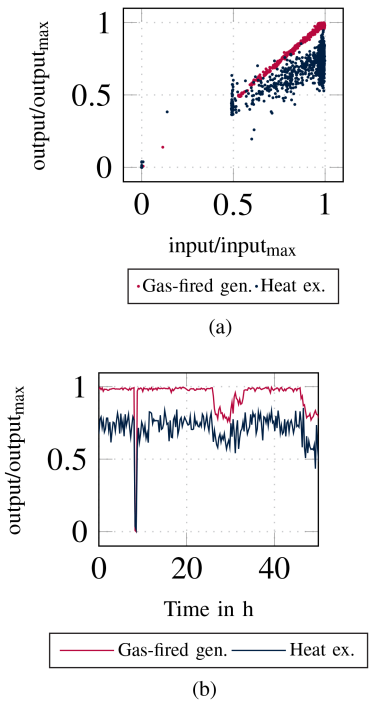


FIGURE 9. Visualization of preprocessed time series data. (a) Input–output. (b) Output-time.

must be in operation for 8.15 h. As described in Algorithm 6, for durations greater than a threshold value, maximum holding durations are set to ∞, i.e., no limits are enforced during the optimization. The set of follower states contains the different state in each case, which allows any state to be reached.

Parameters for ramp limits have also been derived. Both states have lower ramp limits of 0 and (very) high upper limits.

TABLE 5. Parameters Extracted From Preprocessed Time Series Data (P1 and P2)

Subset	Parameter	Gas-fired gen.	Heat ex.
P1 Operational boundaries (input)	lower bound, kW	0	0
	upper bound, kW	2912	1101
P1 Operational boundaries (output)	lower bound, kW	0	0
	upper bound, kW	1101	1746
P2 IO-Relationship	slope, kW/kW	0.38	1.15
	intercept, kW	-18.4	55.2

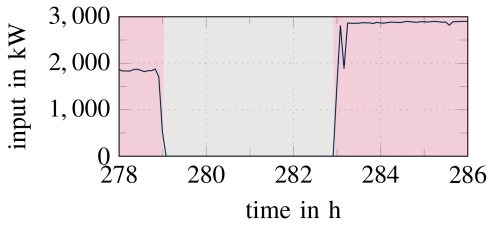


FIGURE 10. Detected system states of resource 1.

TABLE 6. System States and Related Parameters of the Gas-Fired Generator (P3)

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	∞

This means that the resource is capable of rapidly changing power setpoints.

B. SYSTEM SPECIFIC PARAMETERS

This section shows the results of deriving the operational boundaries of the system as well as the selection of a temporal resolution for the optimization model.

Table 7 shows the operational boundaries of the system, as shown in Fig. 7, derived by means of the algorithm described in Section IV-E1. The parameters align with the parameters given in Table 5.

As described in Section IV-E2, the selection of a suitable temporal resolution is also necessary. By means of Algorithm 12, a temporal resolution for the optimization model of 0.125 h has been selected. The holding durations have been converted accordingly from time to time steps, e.g.,

TABLE 7. Operational Boundaries of the System

Parameter	Input (E Gas)	Output (E El.)	Output (E Th.)
lower bound, kW	0	0	0
upper bound, kW	2912	1101	1746

TABLE 8. Dependencies of Resources Within the CHP System (see Fig. 7)

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

the minimum holding duration of resource 1 in state 0 has been converted from 0.5 h to four time steps.

C. EXTRACTION OF DEPENDENCY INFORMATION

This section describes the application of the algorithm described in Section IV-D2 for the extraction of dependencies from the FPD information model of the system shown in Fig. 7. Table 8 gives the list of dependencies present within the system.

To further demonstrate the applicability of Algorithms 9–11, the dependencies of the example systems, as shown in Fig. 5, have been extracted. This includes the demonstration that both dependency types can be extracted. The dependencies are given in Table 9.

D. CASE STUDY

Following the derivation of parameters, the extraction of dependency information, and their persistence to the data model, this section describes the parameterization of an optimization model, following the modeling approach described in Section II, of the system of flexible energy resources depicted in Fig. 7. This is accomplished by applying the design patterns for flexible energy resource optimization models presented by Wagner et al. [7]. By parameterizing the generic model structure, a solvable instance of the optimization model for the system is created. The optimization model is then used to validate the parameters against time series data columns not previously used to derive the parameters ("validation dataset").

As the aim of this validation is not the generation of an optimized schedule but the validation of the algorithms for the derivation of parameters, which have been used to generate the system parameter set displayed in Tables 5–8, the model is solved as a simulation model. Thus, no objective function is necessary. To ensure comparability, the system input time series data column is equated to the corresponding decision variable for the input flow of the system. The selection of system states and the related constraints (holding durations per state) impacts the resulting schedule, as the corresponding

TABLE 9. Dependencies Extracted From the FPD Information Model (see Fig. 5)

Dependency	type	output resources	input resources
Fig. 5b-1	correlative	system input	Resource 1
Fig. 5b-2	correlative	Resource 1	Resource 2
Fig. 5b-3	correlative	Resource 2	system output
Fig. 5c-1	restrictive	system input	Resource 1, Resource 2
Fig. 5c-2	correlative	Resource 1, Resource 2	system output

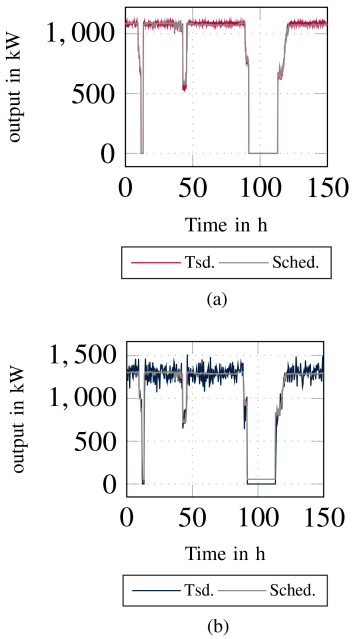


FIGURE 11. Comparison of generated schedules (Sched.) with corresponding time series data columns (Tsd.). (a) Output of gas-fired gen. (b) Output of heat ex.

constraints must be met. As described in Section II, the constraints include the selection of system states, fulfillment of holding durations, ramp limits, the input–output relationship, operational boundaries, and dependencies [7].

The alignment of the resources' outputs with the corresponding time series data columns can be assessed by means of the root mean square error (RMSE). The normalized root mean square error (nRMSE) can be calculated from the RMSE by division with the maximum value of the preprocessed time series data column.

As the input of the gas-fired generator was equated to the corresponding time series data column, the nRMSE is 0%. Fig. 11(a) shows the alignment of generated schedule for the output of the gas-fired generator and the corresponding time series data column. The RMSE is 12.5 kW (nRMSE: 1%). Analogously, Fig. 11(b) shows the comparison of the output of the heat exchanger. The RMSE of the heat exchanger model is 92 kW (nRMSE: 6%). This is due to the fluctuations in the time series data column of the heat exchanger [as can be

seen in Figs. 9 and 11(b)]. In addition, as the heat exchanger does not exhibit system states and the intercept of the model is positive (see Table 5), an input of zero still results in an output of 55.2 kW. A nRMSE of 6% is still within the typical range found in the state of the art [60].

Furthermore, as shown in Fig. 7 and Table 8, the output of the gas-fired generator acts as the input of the heat exchanger. Thus, any inaccuracies of the model of the gas-fired generator directly impact the schedule of the heat exchanger.

The alignment of the generated schedules with the corresponding time series data columns, as visualized in Fig. 11, demonstrates the applicability and reliability of the methodology for the data preprocessing, numerical parameter derivation, and dependency extraction described in Section IV.

E. COMPARISON OF METHODOLOGY AND ITS RESULTS TO RELATED WORK

This section provides a comparison of the methodology described in Section IV and the results reported in Sections V-A–V-C to methods presented in related work referenced in Section III.

The methodology enhances the parameter derivation process for optimization models by incorporating advanced data preprocessing steps. These steps utilize statistical methods to assess the significance and stationarity of data and replace erroneous values (requirements ② and ③), thus bolstering the reliability of the derived parameters. Such preprocessing is crucial in addressing the gaps identified in related works, where often only partial preprocessing measures are applied [12], [14].

A key feature of this methodology is its utilization of a data model designed to persist parameters, providing a robust foundation for resource modeling. This model integrates numerical parameters with information on resource dependencies, facilitating a comprehensive approach to the modeling of various systems of resources.

Automated and complete identification of parameters marks a significant advancement of this methodology. Operational boundaries (requirement ④), input–output relationships (requirement ⑤), system states and related parameters (requirement ⑥), and storage parameters (requirement ⑦) are identified through data-driven modeling and system identification techniques. This automated process contrasts with the manual efforts described in related work, such as [38], [39], and [40], streamlining the parameterization process significantly.

Furthermore, this methodology goes beyond merely extracting numerical parameters by incorporating information on the system's structure. It achieves a comprehensive understanding of both the relationships between resources (requirement ⑧) and the types of these dependencies (requirement ⑨), addressing an area where previous methods have focused primarily on connectivity without adequately analyzing dependency types [17], [18], [23].

In summary, this methodology contributes significantly to the development of more precise and robust optimization models by merging advanced preprocessing with representation of system structures.

VI. DISCUSSION

The methodology introduced in this work demonstrates robustness and efficacy in describing resource behavior through a comprehensive set of parameters. By integrating advanced data preprocessing and parameter derivation techniques, this methodology ensures high-quality input data and parameter accuracy, leading to high fidelity in optimization models.

The methodology enhances the parameterization process by automating the identification of operational boundaries, input–output relationships, and system states. Such automation not only improves the model's precision but also significantly reduces the potential for human error, which is often a limiting factor in manual parameter identification methods.

The optimization model parameterized through this methodology produces schedules that closely align with corresponding time series data, underscoring the practical applicability and reliability of the approach. Despite the complexity involved in managing error propagation, the model achieved an nRMSE of 1% and 6%, respectively, which is within the acceptable range of the state of the art. This accuracy in representing resource behavior highlights the effectiveness of the methodology in managing and compensating for potential errors across connected modules.

A sufficiently long interval of resource operation should be utilized for the application of this methodology to ensure that the time series data used for deriving numerical parameters encompass all possible and desired resource behaviors.

Errors may propagate when using algorithms to derive parameters for multiple resources within a system, potentially affecting the operational behavior of dependent resources. To this end, and to minimize deviations between optimized and feasible schedules, maintaining a low nRMSE is critical. The crucial role of Algorithm 1 in ensuring the statistical significance of the dataset is fundamental. By rigorously preprocessing the data, this algorithm ensures that the foundational dataset used for parameter derivation is of high quality and statistically significant, laying the groundwork for highly accurate parameter derivation, even for those parameters that are dependent on other previously derived parameters, such as system state relate parameters, e.g., holding durations.

The algorithms for the derivation of dependent parameters, specifically Algorithms 4–8, are designed robustly. Algorithm 4 independently calculates state boundaries for each state, Algorithm 5 ensures a complete mapping of follower states without redundancy, and Algorithm 6 applies dynamic thresholding to distinguish meaningful state durations. Algorithm 7 accurately calculates slopes within system states to reflect transient dynamics, benefiting from the preprocessing and thus providing precise dynamic insights. Algorithm 8 robustly determines ramp limits by analyzing slopes of state transitions, adapting to temporal changes. Together, these algorithms exhibit adaptability, precision, and

comprehensive data analysis, ensuring accurate parameter estimation for reliable system optimization.

VII. CONCLUSION

This work presents a methodology for the automated derivation of parameters of optimization models of systems of flexible energy resources. For this purpose, a data model for the persistence of optimization model parameters has been created. Numerical parameters are derived from time series data after preprocessing within an integrated methodology. Dependencies, i.e., connections of flows of individual resources within the system, are extracted from an information model representing the structure of the system.

The methodology addresses the previously labor-intensive and error-prone nature of parameter derivation. Through the structured process of parameter derivation, it ensures the accuracy of parameters. Consequently, utilizing a validated model structure, such as the design patterns [7], leads to accurate optimization results. This structured approach significantly contributes to the development of more accurate and robust optimization models by combining advanced preprocessing with a comprehensive representation of system structures.

The validation shows the applicability of the methodology for the parameterization of an optimization model of a CHP system consisting of two resources. The use of the parameters in an optimization model of the system for the generation of a schedule and its comparison with previously not used time series shows good agreement with low RMSE values of 1% (gas-fired generator) and 6% (heat exchanger). The methodology has been validated by means of a specific type of flexible energy resource. Various kinds of flexible energy resources, such as battery energy storage systems, electrolyzers, heat pumps, or consumers [6], can be represented by a parameterized instance of the generic optimization model structure [7]. As the model structure represents resources within systems by representing the operational boundaries, the relationship of their input and output as well as their system states, this modeling concept [7] can also be applied for the optimization of a multitude of resource types. Thus, the methodology described in this work can also be widely applied.

In future work, the data preprocessing could be extended by the method of Roche et al. [61], which applies autoencoders and AutoDiff to reconstruct missing data in time series. This would yield increased data quality. In addition to the parameterization of optimization models, the method for deriving parameters could also be applied to for the creation of energy performance indicator models [51], which currently poses challenges to practitioners.

ACKNOWLEDGMENT

The authors would like to thank Nemanja Hranisavljevic for his input on algorithms for the data-based identification of system states. Appreciation is also extended to Tom Westermann for his assistance in implementing the clustering algorithm. In addition, the authors are thankful to the four anonymous reviewers for their constructive feedback.

REFERENCES

[1] IRENA and European Comission, "Renewable energy prospects for the European union: Preview for policy makers," 2018, Accessed: Feb. 1, 2024. [Online]. Available: https://www.irena.org/publications/2018/Feb/Renewable-energy-prospects-for-the-EU

[2] Agora Energiewende, Prognos, Consentec, "Climate-neutral power system 2035 (ger.: Klimaneutrales Stromsystem 2035): How the German power sector can become climate-neutral by 2035," 2022. [Online]. Available: https://static.agora-energiewende.de/fileadmin/Projekte/2021/2021_11_DE_KNStrom2035/A-EW_264_KNStrom2035_WEB.pdf

[3] A. Pfendler et al., "Vision of future energy (ger.: Zukunftsbild energie): VDE study," 2022. [Online]. Available: https://www.vde.com/resource/blob/2228542/eebdadd7a64b5d00b18db8e1c8262377/zukunftsbild-energie-download-data.pdf

[4] L. M. Reinpold, L. P. Wagner, M. Kilthau, A. Karmann, and A. Fay, "Planning Functions for (energy) storage systems," *atp magazin*, vol. 65, no. 3, pp. 60–69, 2023, doi: 10.17560/atp.v65i3.2652.

[5] A. Ulbig and G. Andersson, "On operational flexibility in power systems," in *Proc. IEEE 2012 Power Energy Soc. Gen. Meeting*, San Diego, CA, USA, 2012, pp. 1–8, doi: 10.1109/PESGM.2012.6344676.

[6] L. P. Wagner, L. M. Reinpold, M. Kilthau, and A. Fay, "A systematic review of modeling approaches for flexible energy resources," *Renewable Sustain. Energy Rev.*, vol. 184, 2023, Art. no. 113541, doi: 10.1016/j.rser.2023.113541.

[7] L. P. Wagner, L. M. Reinpold, and A. Fay, "Design patterns for optimization models of flexible energy resources," in *Proc. IEEE 2nd Ind. Electron. Soc. Annu. Online Conf. (ONCON)*, 2023, pp. 1–6, doi: 10.1109/ONCON60463.2023.10430916.

[8] L. Kasper, P. Schwarzmayr, F. Birkelbach, F. Javernik, M. Schwaiger, and R. Hofmann, "A digital twin-based adaptive optimization approach applied to waste heat recovery in green steel production: Development and experimental investigation," *Appl. Energy*, vol. 353, 2024, Art. no. 122192, doi: 10.1016/j.apenergy.2023.122192.

[9] M. Runge, L.-T. Reiche, K.-H. Niemann, and A. Fay, "Universal energy information model for industrial communication," in *Proc. IEEE 2022 27th Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Stuttgart, Germany, 2022, pp. 1–8, doi: 10.1109/ETFA52439.2022.9921557.

[10] P. Schott, J. Sedlmeir, N. Strobel, T. Weber, G. Fridgen, and E. Abele, "A generic data model for describing flexibility in power markets," *Energies*, vol. 12, no. 10, 2019, Art. no. 1893, doi: 10.3390/en12101893.

[11] R. Bahmani, C. van Stiphoudt, S. P. Menci, M. Schöpf, and G. Fridgen, "Optimal industrial flexibility scheduling based on generic data format," *Energy Informat.*, vol. 5, no. S1, 2022, Art. no. 26, doi: 10.1186/s42162-022-00198-4.

[12] Y. Choi et al., "Time-series clustering approach for training data selection of a data-driven predictive model: Application to an industrial bio 2,3-butanediol distillation process," *Comput. Chem. Eng.*, vol. 161, 2022, Art. no. 107758, doi: 10.1016/j.compchemeng.2022.107758.

[13] X. Luo, T. Hong, Y. Chen, and M. A. Piette, "Electric load shape benchmarking for small- and medium-sized commercial buildings," *Appl. Energy*, vol. 204, pp. 715–725, 2017, doi: 10.1016/j.apenergy.2017.07.108.

[14] R.-H. Hechelmann, "Predictive simulation-based optimization of a refrigeration supply system (ger.: Prädiktive simulationsgestützte Optimierung eines Kälteversorgungssystems)," Ph.D. thesis, Univ. Kassel, Kassel, Germany, 2021.

[15] H. Li, Z. Wang, T. Hong, A. Parker, and M. Neukomm, "Characterizing patterns and variability of building electric load profiles in time and frequency domains," *Appl. Energy*, vol. 291, 2021, Art. no. 116721, doi: 10.1016/j.apenergy.2021.116721.

[16] M. Barbero, V. Rebillas-Loredo, R. Valdés, and C. Corchero, "Data-driven demand flexibility estimation in a commercial buildings from air conditioning and lighting system," in *Proc. World Sustain. Energy Days 2021 (WSED): 23rd Eur. Photovol. Sol. Energy Conf.: Wels, Austria*, 2021, pp. 1–7. [Online]. Available: http://hdl.handle.net/2117/352859

[17] T. Jeleniewski, J. Reif, and A. Fay, "Integrating interdependencies in semantic manufacturing process description models," in *Proc. 2023 IEEE 28th Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sinaia, Romania, 2023, pp. 1–4, doi: 10.1109/ETFA54631.2023.10275608.

[18] S. Y. Yim, H. G. Ananthakumar, L. Benabbas, A. Horch, R. Drath, and N. F. Thornhill, "Using process topology in plant-wide control loop performance assessment," *Comput. Chem. Eng.*, vol. 31, no. 2, pp. 86–99, 2006, doi: 10.1016/j.compchemeng.2006.05.004.

[19] X.-L. Hoang and A. Fay, "A capability model for the adaptation of manufacturing systems," in *Proc. IEEE 2019 24th Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Zaragoza, Spain, 2019, pp. 1053–1060, doi: 10.1109/ETFA.2019.8869142.

[20] "Engineering data exchange format for use in industrial automation systems engineering - Automation Markup Language - Part 1: Architecture and general requirements," IEC 62714–1:2018, 2018.

[21] "Formalised process descriptions-Concept and graphic representation," VDI/VDE 3682 Part 1:2015-05, 2015.

[22] M. Hoernicke, C. Messinger, E. Arroyo, and A. Fay, "Topology models in AutomationML–Object-oriented basis for the automation of automation," atp edition, vol. 58, 2016, Art. no. 28, doi: 10.17560/atp.v58i05.565.

[23] J. Thambirajah, L. Benabbas, M. Bauer, and N. F. Thornhill, "Cause-and-effect analysis in chemical processes utilizing XML, plant connectivity and quantitative process history," *Comput. Chem. Eng.*, vol. 33, no. 2, pp. 503–512, 2009, doi: 10.1016/j.compchemeng.2008.10.002.

[24] M. K. Petersen, K. Edlund, L. H. Hansen, J. Bendtsen, and J. Stoustrup, "A taxonomy for modeling flexibility and a computationally efficient algorithm for dispatch in smart grids," in *Proc. 2013 Amer. Control Conf.*, Washington, DC, 2013, pp. 1150–1156, doi: 10.1109/ACC.2013.6579991.

[25] N. Wanapinit, J. Thomsen, C. Kost, and A. Weidlich, "An MILP model for evaluating the optimal operation and flexibility potential of end-users," *Appl. Energy*, vol. 282, 2021, Art. no. 116183, doi: 10.1016/j.apenergy.2020.116183.

[26] M. T. Baumhof, E. Raheli, A. G. Johnsen, and J. Kazempour, "Optimization of hybrid power plants: When is a detailed electrolyzer model necessary?," in *Proc. IEEE 2023 PowerTech*, Belgrade, Serbia, 2023, pp. 1–10, doi: 10.1109/PowerTech55446.2023.10202860.

[27] L. F. J. Barth, N. N. Ludwig, E. Mengelkamp, and P. Staudt, "A comprehensive modelling framework for demand side flexibility in smart grids," *Comput. Sci. - Res. Develop.*, vol. 33, no. 1-2, pp. 13–23, 2018, doi: 10.1007/s00450-017-0343-x.

[28] H. Li and T. Hong, "A semantic ontology for representing and quantifying energy flexibility of buildings," *Adv. Appl. Energy*, vol. 8, 2022, Art. no. 100113, doi: 10.1016/j.adapen.2022.100113.

[29] A. Fernandez-Izquierdo, A. Cimmino, and R. Garcia-Castro, "Supporting demand-response strategies with the DELTA ontology," in *Proc. 2021 IEEE/ACS 18th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Tangier, Morocco, 2021, pp. 1–8, doi: 10.1109/AICCSA53542.2021.9686935.

[30] B. Biegel, P. Andersen, J. Stoustrup, L. H. Hansen, and D. V. Tackie, "Information modeling for direct control of distributed energy resources," in *2013 Amer. Control Conf.*, Washington, DC, USA, 2013, pp. 3498–3504, doi: 10.1109/ACC.2013.6580372.

[31] "Energy management system application program interface (EMS-API)," IEC 61970-1:2005, 2005.

[32] "Application integration at electric utilities–System interfaces for distribution management – Part 5: Distributed energy optimization," FprEN IEC 61968-5:2020, 2020.

[33] D. A. Dickey and W. A. Fuller, "Distribution of the estimators for autoregressive time series with a unit root," *J. Amer. Stat. Assoc.*, vol. 74, no. 366, 1979, Art. no. 427, doi: 10.2307/2286348.

[34] E. L. Lehmann and J. P. Romano, "Uniformly most powerful tests," in *Testing Statistical Hypotheses*, E. L. Lehmann and J. P. Romano, Eds., Cham, Switzerland: Springer International Publishing, 2022, pp. 61–124.

[35] E. M. Olariu, R. Tolas, R. Portase, M. Dinsoreanu, and R. Potolea, "Modern approaches to preprocessing industrial data," in *Proc. IEEE 2020 16th Int. Conf. Intell. Comput. Commun. Process.*, Cluj-Napoca, Romania, 2020, pp. 221–226, doi: 10.1109/ICCP51029.2020.9266215.

[36] A. A. Bachnas, R. Tóth, J. Ludlage, and A. Mesbah, "A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study," *J. Process Control*, vol. 24, no. 4, pp. 272–285, 2014, doi: 10.1016/j.jprocont.2014.01.015.

[37] L. Ljung, "Perspectives on system identification," *Annu. Rev. Control*, vol. 34, no. 1, pp. 1–12, 2010, doi: 10.1016/j.arcontrol.2009.12.001.

[38] P. A. Gade, T. Skjøtskift, C. Ziras, H. W. Bindner, and J. Kazempour, "Load shifting versus manual frequency reserve: Which one is more appealing to thermostatically controlled loads in Denmark?," *Electric Power Syst. Res.*, vol. 232, 2024, Art. no. 110364, doi: 10.1016/j.epsr.2024.110364.

[39] L. M. Reinpold, L. P. Wagner, L.-T. Reiche, and A. Fay, "Experimental setup for the evaluation of optimization strategies for flexible energy resources," in *Proc. IEEE 2nd Ind. Electron. Soc. Annu. Online Conf. (ONCON)*, 2023, pp. 1–6, doi: 10.1109/ONCON60463.2023.10430717.

[40] B. Flamm, C. Peter, F. N. Büchi, and J. Lygeros, "Electrolyzer modeling and real-time control for optimized production of hydrogen gas," *Appl. Energy*, vol. 281, 2021, Art. no. 116031, doi: 10.1016/j.apenergy.2020.116031.

[41] H. Li and T. Hong, "On data-driven energy flexibility quantification: A framework and case study," *Energy Buildings*, vol. 296, 2023, Art. no. 113381, doi: 10.1016/j.enbuild.2023.113381.

[42] R.-H. Peesel, F. Schlosser, H. Meschede, H. Dunkelberg, and T. Walmsley, "Optimization of cooling utility system with continuous self-learning performance models," *Energies*, vol. 12, no. 10, 2019, Art. no. 1926, doi: 10.3390/en12101926.

[43] J. Liisberg, J. K. Møller, H. Bloem, J. Cipriano, G. Mor, and H. Madsen, "Hidden Markov models for indirect classification of occupant behaviour," *Sustain. Cities Soc.*, vol. 27, pp. 83–98, 2016, doi: 10.1016/j.scs.2016.07.001.

[44] T. Westermann, M. S. Gill, and A. Fay, "Representing timed automata and timing anomalies of cyber-physical production systems in knowledge graphs," in *Proc. IECON 2023- 49th Annu. Conf. IEEE Ind. Electron. Soc.*, Singapore, Singapore, 2023, pp. 1–7, doi: 10.1109/IECON51785.2023.10312156.

[45] N. N. Ludwig, S. Waczowicz, R. Mikut, and V. Hagenmeyer, "Assessment of unsupervised standard pattern recognition methods for industrial energy time series," in *Proc. 9th Int. Conf. Future Energy Syst.*, Karlsruhe Germany, 2018, pp. 434–435, doi: 10.1145/3208903.3212051.

[46] "Formalised process descriptions-information model," VDI/VDE 3682 part 2:2015-05, 2015.

[47] T. Jäger, L. Christiansen, M. Strube, and A. Fay, "Integrated engineering from requirements elicitation to the plant structure description," *at - Automatisierungstechnik*, vol. 61, no. 2, pp. 92–101, 2013, doi: 10.1524/auto.2013.0010.

[48] H. Nabizada, A. Köcher, C. Hildebrandt, and A. Fay, "Open, web-based tool for information modeling with formalized process description (ger.: Offenes, webbasiertes Werkzeug zur Informationsmodellierung mit Formalisierter Prozessbeschreibung)," in *Proc. 21 VDI-Kongress AUTOMATION, VDI e. V., Ed.*, Baden-Baden, 2020, pp. 443–454, doi: 10.51202/9783181023754-443.

[49] X.-L. Hoang, P. Marks, M. Weyrich, and A. Fay, "Modeling of interdependencies between products, processes and resources to support the evolution of mechatronic systems," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4348–4353, 2017, doi: 10.1016/j.ifacol.2017.08.873.

[50] M. Barth and A. Fay, "Automated generation of simulation models for control code tests," *Control Eng. Pract.*, vol. 21, no. 2, pp. 218–230, 2013, doi: 10.1016/j.conengprac.2012.09.022.

[51] "ISO 50006: 2023-05 Energy management systems - Evaluating energy performance using energy performance indicators and energy baselines," General principles and guidance.

[52] L. P. Wagner, M. Kilthau, L. M. Reinpold, and A. Fay, "Required level of detail of optimization models for the control of flexible energy resources," in *Proc. IEEE 2023 Int. Conf. Commun., Control, Comput. Technol. Smart Grids*, 2023, pp. 1–6, doi: 10.1109/SmartGridComm57358.2023.10333938.

[53] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques* (The Morgan Kaufmann Series in Data Management Systems Series). Amsterdam, The Netherlands: Kaufmann and Elsevier Science, 2005.

[54] C. M. Bishop, *Pattern Recognition and Machine Learning* (Computer Science Series). New York, NY, USA: Springer, 2006.

[55] ILOG CPLEX, "ILOG CPLEX 11.0 user's manual: Numeric difficulties," 2007. [Online]. Available: https://www-eio.upc.es/lceio/manuals/cplex-11/html/usrcplex/solveLP17.html

[56] M. H. Satman and A. Kopilov, "RCaller: A java package for interfacing R," *J. Open Source Softw.*, vol. 5, no. 55, 2020, Art. no. 2722, doi: 10.21105/joss.02722.
[57] V. M. R. Muggeo, "Estimating regression models with unknown break-points," *Statist. Med.*, vol. 22, no. 19, pp. 3055–3071, 2003, doi: 10.1002/sim.1545.
[58] Accessed: May 6, 2024. "Gson," [Online]. Available: https://github.com/google/gson
[59] V. Henkel, L. P. Wagner, F. Gehlhoff, and A. Fay, "Combination of site-wide and real-time optimization for the control of systems of flexible energy resources," 2024. [Online]. Available: http://arxiv.org/pdf/2404.06748
[60] E. Raheli, Y. Werner, and J. Kazempour, "Flexibility of integrated power and gas systems: Modeling and solution choices matter," 2023. [Online]. Available: https://arxiv.org/pdf/2311.05744
[61] J.-P. Roche, O. Niggemann, and J. Friebe, "Using autoencoders and AutoDiff to reconstruct missing variables in a set of time series," 2023. [Online]. Available: https://arxiv.org/pdf/2308.10496

LUKAS PETER WAGNER received the M.Eng. degree in energy and environmental management from the University of Flensburg, Flensburg, Germany, in 2020.

He is currently a Research Associate with the Institute of Automation Technology, Helmut Schmidt University Hamburg, Germany. His research interests include the modeling and optimization of flexible energy resources.

ALEXANDER FAY (Senior Member, IEEE) received the Diploma and Ph.D. degrees in electrical engineering from the Technical University of Braunschweig, Braunschweig, Germany, in 1995 and 1999, respectively.

He had worked for five years with ABB Corporate Research Laboratories in Heidelberg and Ladenburg before he was appointed as a Full Professor with the Institute of Automation, Helmut Schmidt University, Hamburg, Germany, in 2004. He currently holds the Chair of Automation, Ruhr University, Bochum, Germany. His research interests include models, methods, and tools for the engineering of large-scale automated systems, such as production plants, buildings, logistics systems, and energy systems.

Dr. Fay was the Member for the AdCom of the IEEE Industrial Electronics Society. He was the General Co-Chair of *IEEE International Conference on Emerging Technologies and Factory Automation* (ETFA), in 2008, and the Program Co-Chair of *IEEE International Conference on Automation Science and Engineering* (CASE), in 2018. From 2011 to 2017, he was an Associate Editor for IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS.