Open Source Simulation Software pandapower and pandapipes: Recent Developments

Roman Bolgaryn*, Gourab Banerjee*§, Steffen Meinecke*§ Hendrik Maschke*, Frank Marten*, Mario Richter*, Zheng Liu§, Pawel Lytaev§, Baraa Alfakhouri§, Jolando Kisse§, Daniel Lohmeier*‡
*Division of Grid Planning and Operation, Fraunhofer IEE, Kassel, Germany
§Department of Energy Management and Power System Operation, University of Kassel
‡retoflow GmbH, Kassel, Germany
roman.bolgaryn@iee.fraunhofer.de

2023 Open Source Modelling and Simulation of Energy Systems (OSMSES) | 979-8-3503-1122-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/OSMSES58477.2023.10089685

***Abstract*—`pandapower` and `pandapipes` are open source Python libraries that were developed to analyze electric and pipe grids (i.e. natural gas, hydrogen, water, and district heating networks), respectively. In the last year, there have been new developments in `pandapower` and `pandapipes`. We describe the most important new features that were added after the release of `pandapower` version 2.6.0 and `pandapipes` version 0.4.0. This paper refers to `pandapower`'s latest version 2.12.0 and `pandapipes`' latest version 0.8.4 and is organized as follows. First, we introduce the new features of `pandapower`. We added a new module to simulate protection devices. Furthermore, we added the implementation of static grid equivalents. Groups of elements will be useful to modify or obtain results for several elements at once. In addition, we extended the power flow calculation in `pandapower` by two Flexible AC Transmission System (FACTS) devices and by Temperature-Dependent Power Flow (TDPF). We further improved the short-circuit calculation and added voltage as a result in order to enable the implementation of distance protection in the future. Finally, we added a converter from `DigSILENT PowerFactory` to `pandapower`. A further converter we added can be used to import a CIM CGMES dataset to pandapower. As for `pandapipes`, we improved its performance through just-in-time compiled functions. Furthermore, new component types were added, including pressure regulator, flow control, and compressor. Finally, a converter from `STANET` to `pandapipes` was added.**

***Index Terms*—power system modeling, power system protection, grid calculation, automation, short circuit calculation, temperature-dependent power flow, FACTS devices, gas grid modeling, district heating**

I. Introduction

The open source software `pandapower` enables automated analysis and optimization of electrical power systems. Similarly, `pandapipes` was developed to perform pipe grid simulations. Both libraries rely on the Python package *pandas* [1] for their internal data structure and functionalities. Due to the transformation of the energy system towards more renewable generation as well as the electrification of mobility and heating sectors, the need for adaptation in grid planning and operation rises. Consequently, the emerging complexity needs to be addressed with fitting simulation software, for example by automation of workflows enabled by Python. Open source software such as `pandapower` and `pandapipes` can be further expanded and modified by the open-source community to accommodate additional use cases. Furthermore, scientific excellence and collaboration are beneficial for development and expansion. Moreover, these software libraries are free, user-friendly, transparent, and validated by matching the results of reference calculations with established commercial software. The development of `pandapower` and `pandapipes` is continuously ongoing and documented [2]–[4]. The latest changes in these two Python libraries are described in the following sections.

II. Open Source Software `pandapower`

A. Extending the short-circuit analysis

We extend the existing implementation of the short-circuit analysis with the calculation of current phasors and voltage phasors. This extension allows obtaining the magnitude and angle values for the voltage and current and facilitates including the active and reactive short-circuit power in the results of a short-circuit analysis. To develop this feature, we adjust the existing `pandapower` implementation of the equivalent voltage source method by evaluating the real and imaginary components of the parameters.

This enables the implementation of protection devices, e.g. directional overcurrent relays and distance relays. In this release of `pandapower`, non-directional overcurrent protection is implemented and described in the following section. Further protection devices will be included in the following releases.

B. Power system protection

A new module is introduced to `pandapower` to study and analyze the power system protection behavior for the transmission and distribution grid. The conventional overcurrent relay models are implemented and developed in `pandapower` based on German VDE technical manuals [5], [6].

*1) **Protection framework**:* Based on the main functionalities of protection devices, the implemented relay model has three functions as follows:

- Setting calculation
- Measurement
- Trip decision

The functions of the relay model are introduced in Fig. 1. In the first function, the characteristic parameters of the relay

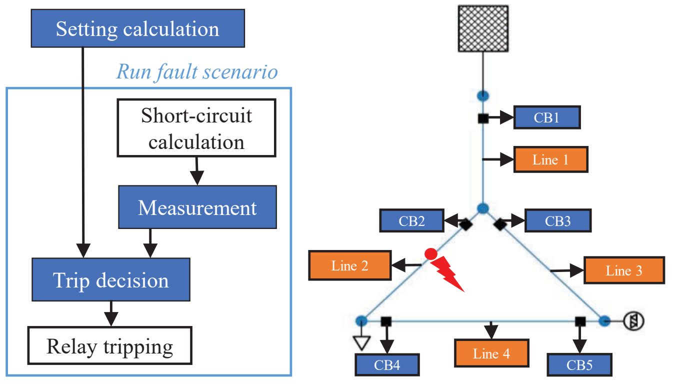

Fig. 1. Left: general protection module architecture, right: fault scenario example

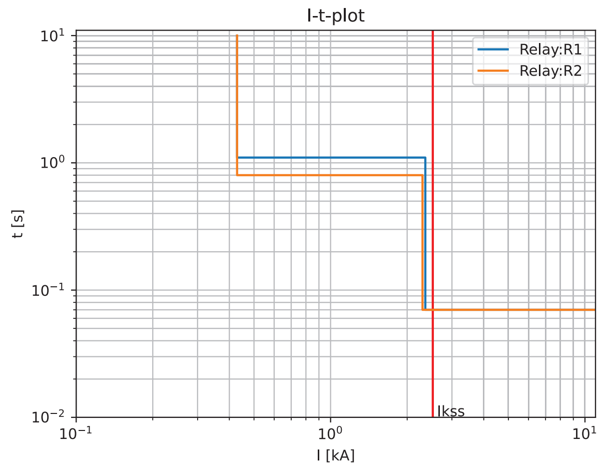

Fig. 2. Example I-t-curve for two DTOC relays

model are calculated. These include primary relay current threshold ($I_{\gg}$), backup relay current threshold ($I_{>}$), the primary tripping time ($t_{\gg}$) and backup tripping time ($t_{>}$). The parameter settings depend on the type of relay used. The calculated settings are passed through the fault scenario function, comprising a short-circuit calculation, the measurement function and the trip decision function (Fig. 1, left).

After the relays are parameterized, a fault event is simulated at a selected line. The fault location is represented internally by a new auxiliary bus at the selected line length ratio. Next, the fault is simulated via a short-circuit calculation for the auxiliary bus. The fault location and the positions of the relays for an exemplary power system are shown in Fig. 1, right.

The measurement function simulates the behavior of the current transformer (CT) and the voltage transformer (VT). The measurements of the CT and VT are processed into an appropriate signal, e.g., the current at the location of the relay in case of overcurrent protection. The signal is then passed on to the trip decision function.

The trip decision function receives two inputs, namely the output of the setting calculation function and the outputs of the measurement function. The measurements are compared with the thresholds defined in the relay settings and a trip signal is generated. The trip signal is the result of the comparison between the measurement results and the relay settings. Apart from the Boolean trip signal, the tripping time is given as a result of the trip decision function. In the current stage of protection model implementation, the resulting tripping time is displayed to the user as a static value. The given relay functions can be incorporated into a `pandapower` controller with time-series grid data. The measurement and trip decision functions are to be incorporated into the *is_converged* method and the resulting relay tripping into the *control_step* function of the `pandapower` controller. Then the tripping time can be represented on a real-time basis during the time series simulation by opening the breakers at the relay locations according to the given tripping schedule.

2) ***Overcurrent relay implementation:*** In the recent `pandapower` contributions, this universal protection modeling workflow is applied to the definite time overcurrent (DTOC) and inverse definite minimum time (IDMT) relays.

TABLE I
CURRENT THRESHOLDS CALCULATION

Setting	Calculation	Description
$I_{>}$	$1.5{\cdot}I_n$ (rated line current)	line overloading factor $(1.2)\cdot$ CT current factor $(1.25)\cdot I_n$
$I_{\gg}$	I_{sc} (three-phase short-circuit current)	minimum short-circuit current for faults at the end of the faulty line

The setting calculations are based on the current VDE protection standards and best practices from literature [7], [8]. We provide default threshold settings in Table I, which are based on [9] and can be adjusted by the user if necessary.

In the DTOC relay, an automated setting calculation function returns $I_{\gg}$ based on the three-phase short-circuit current [10] flowing at the end of the faulty line, which is considered as the pickup current setting of the primary protection. Additionally, $I_{>}$ is calculated based on the rated line current obtained from the line parameters, the CT current factor and the line overloading factor. The corresponding time settings $t_{\gg}$ and $t_{>}$ are based on the topological distance from the external grid connection.

In the IDMT relay model, the thresholds are calculated based on the IEC 60255 standard [11] in which the tripping time $t_{>}$ depends on the measured short-circuit current by following the tripping inverse curve. The standard inverse settings are implemented by default in `pandapower`. The IEC 60255 options ”very inverse”, ”extremely inverse”, and ”long time standard inverse” can be selected.

The trip decision function for overcurrent relays compares the given I and t settings with the measured current and generates a Boolean trip signal. In addition to the trip signal and tripping time, the current threshold, the fault current, and information of instantaneous ($I_{\gg}$) or backup ($I_{>}$) tripping are provided as an output result, which can be used for post-processing actions. The $I-t$ plots represent the relay curve graphically and are useful to interpret the results of the protection simulation. Fig. 2 demonstrates an example DTOC relay curve for two relays R1 and R2 at a fault occurrence,

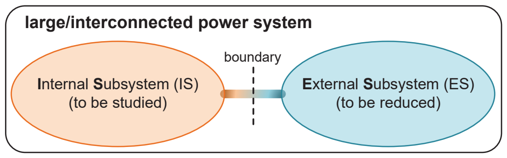

Fig. 3. Illustration of subsystems for grid equivalent

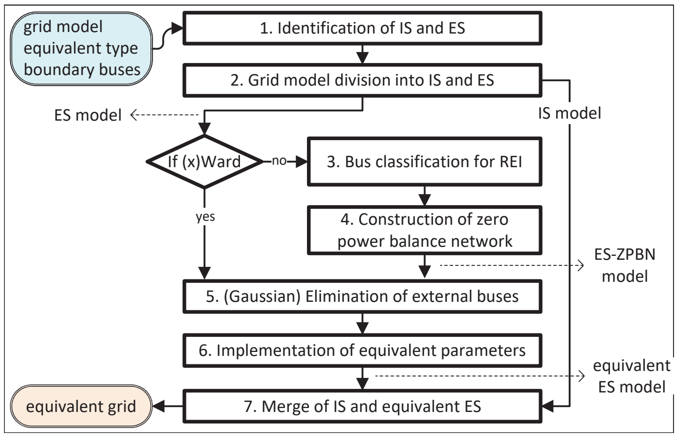
grid model
equivalent type
boundary buses
1. Identification of IS and ES
2. Grid model division into IS and ES
ES model
IS model
If (x)Ward
no
yes
3. Bus classification for REI
4. Construction of zero power balance network
ES-ZPBN model
5. (Gaussian) Elimination of external buses
6. Implementation of equivalent parameters
equivalent ES model
equivalent grid
7. Merge of IS and equivalent ES

Fig. 4. Flow Chart for Ward and REI equivalent calculations in `pandapower`

and shows the selective tripping during a short-circuit event (red vertical line).

C. Static Grid equivalents

Due to the practical limitation of computational resources, grid data confidentiality, and security-relevant reasons [12], the reduced representation of large power systems using static grid equivalent techniques is of great importance for static analysis such as grid operation, planning, and market-oriented studies [13]. Fig. 3 shows the general case of a large power system, which is divided into the internal subsystem (IS) and the external subsystem (ES). IS and ES could belong to the same or different grid operators. The IS, which is of interest to engineers, remains unmodified, and a simple model represents the ES. The reduced interconnected power system should represent the original system as accurately as possible.

Of the available methods, the Ward and REI (i.e., radial, equivalent, independent) equivalents [14]–[16] as well as the more advanced variant of Ward, called extended Ward (xWard), are the most commonly used static grid equivalents in the industry. These classical equivalent methods are implemented as a new module of `pandapower`. Fig. 4 illustrates the implemented grid equivalent calculations.

The required inputs are the grid model to be reduced with the associated power flow results (in particular, the voltage magnitude and angle values at the boundary buses), the equivalent type, and the boundary buses with which IS and ES can be identified. The procedure can be described as follows:

1 Recognition of IS and ES according to the inputs: all buses are divided into three groups: internal buses, boundary buses, and external buses.
2 Decomposition of the grid model into the IS model and the ES model.
3 (If REI) Bus classification for REI: the external buses are further identified and selected for the construction of a zero power balance grid (ZPBN).
4 (If REI) Construction of ZPBN: by default, loads in the ES are aggregated while generators in ES are reserved individually during the construction.
5 Performing Gaussian Elimination: the external buses for (x)Ward or REI equivalents are eliminated using the Gaussian elimination and as a result of the elimination, the equivalent parameters are calculated.
6 Implementation of equivalent parameters: the equivalent devices are created according to the equivalent parameters, while the external buses are removed. The equivalent ES model is constructed.
7 Merge of the IS model and the equivalent ES model: the constructed equivalent ES model and the IS model are connected at the boundary buses.

We compared the results of power flow calculations before and after the calculation of the grid equivalent for several benchmark grids. The static bus voltage deviations were up to 10^{-6} p.u. for standard IEEE benchmark grids such as 9-bus, 39-bus, and 118-bus test systems with over 30 equivalent scenarios. Their applications are introduced in a tutorial[1]. The implemented REI method was validated by comparing the equivalent power flow results and the calculated equivalent parameters with those provided by `DIgSILENT PowerFactory`[2]. The deviations are up to 10^{-6}. In `DIgSILENT PowerFactory`, the Ward and xWard equivalents are calculated based on additional sensitivity analyses which are not open-source and not documented. Therefore, the validations for the implemented Ward and xWard with `DIgSILENT PowerFactory` are currently difficult.

D. Groups of elements

Recent contributions allow the user to combine several elements of a `pandapower` grid model into groups. This improves the clarity and comprehensibility of data. Furthermore, we introduced new functions that can be applied to all elements of a group. This simplifies getting and setting values, such as power and in service information, from multiple grid elements and element types.

Grouping grid elements might be helpful for different use cases such as summarizing substation elements. Likewise, when calculating static grid equivalents as described by Section II-C, new grid elements such as ward, load, shunt impedance and series impedance elements are automatically grouped together.

Similarly, element groups of different types can be used for simulations of virtual power plants (VPPs) and feeder analyses. To exemplify the new simple function of summing power values of all group members, a case of one VPP

[1]https://github.com/e2nIEE/pandapower/blob/develop/tutorials/grid_equivalents.ipynb

[2]DigSILENT PowerFactory https://www.digsilent.de/en/powerfactory.html

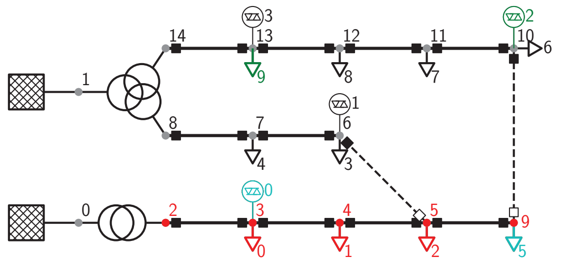

Fig. 5. Case study grid with virtual power plant elements (VPP 1, green) and an area to analyze (Area 1, red). Cyan elements belong to both, VPP1 and Area 1.

TABLE II
DATAFRAME OF GROUP INFORMATION AS STORED IN A PANDAPOWER NET

group index	name	element_type	element
0	VPP 1	sgen	[0, 2]
0	VPP 1	load	[5, 9]
1	Area 1	line	[0, 1, 2, 6]
1	Area 1	switch	[0, 1, . . . , 6, 12, 13, 14]
1	Area 1	sgen	[0]
1	Area 1	load	[0, 1, 2, 5]
1	Area 1	bus	[2, 3, 4, 5, 9]

and one grid area to be analyzed is considered. These two groups are applied to the example grid from the `pandapower` introduction article [2], see Fig. 5. Defining two generation units and two loads as VPP as well as all elements that are part of the lower feeder of the initial grid configuration, the information of the two groups ”VPP 1” and ”Area 1” is stored as a `pandas DataFrame` as shown in Table II.

Executing power flow calculations in a time series simulation using the switch and transformer tap position results of article [2], the group functionality allows accessing easily the sums of VPP 1 and Area 1, as illustrated in Fig. 6.

E. Flexible AC Transmission System elements

We reproduce the implementation of the FACTS devices Static Var Compensator (SVC) and Thyristor Controlled Series Compensator (TCSC) from the Ph.D. dissertation of Ara

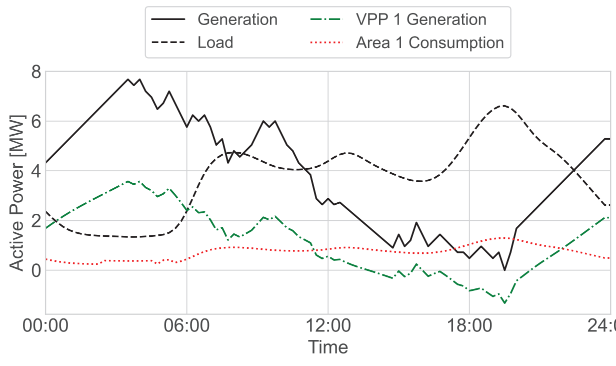

Fig. 6. Results of time series simulation

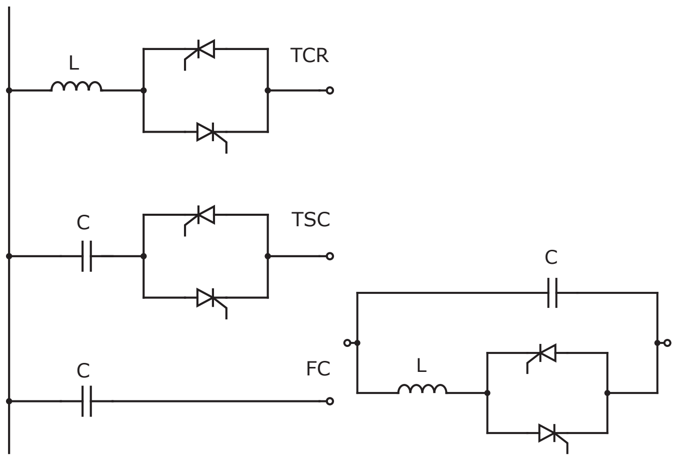

Fig. 7. Circuits of the SVC: a shunt element with the capacitance provided by a TSC and a fixed capacitor (FC), in parallel to a TCR (left), and TCSC: a series element combining a fixed capacitor in parallel to a TCR (right), based on [17], created with [19]

Panosyan [17]. The used method is to formulate a Jacobian modification matrix J_M and add it to the original Jacobian matrix J. The vector of control variables of the FACTS devices x is included in the vector of state variables, and the mismatches are expanded with the mismatch equations for the FACTS devices $\triangle F_x$. Eq. (1) describes obtaining the state variables vector for the next, $k+1$, iteration of the Newton-Raphson method. As a result, the state variables representing the control settings of the FACTS devices (e. g. thyristor firing angle α) are obtained in the control variables vector x, along with the vectors of voltage magnitude V and voltage angle δ, in a single Newton-Raphson power flow calculation.

$$\begin{bmatrix} \delta^{k+1} \\ V^{k+1} \\ x^{k+1} \end{bmatrix} = \begin{bmatrix} \delta^{k} \\ V^{k} \\ x^{k} \end{bmatrix} + (J^k + J_M^k)^{-1} \cdot \begin{bmatrix} \triangle P^k \\ \triangle Q^k \\ \triangle F_x^k \end{bmatrix} \tag{1}$$

*1) **Static Var Compensator**:* SVC is “a shunt-connected static var generator or absorber whose output is adjusted to exchange capacitive or inductive current to maintain or control specific parameters of the electrical power system (typically bus voltage)” [18]. The SVC is a combination of a Thyristor Controlled Reactor (TCR) and a capacitor (fixed, mechanically switched, or Thyristor Switched Capacitor (TSC)) in a parallel configuration, for example as shown in Fig. 7.

The TCR consists of a reactor that is connected in series to two antiparallel thyristors (the top component in the Fig. 7) [17]. The current through the TCR is gradually controlled by adjusting the thyristor firing angle α. Due to the delay of the inductive current of 90° in relation to voltage, the TCR is fully conductive at $\alpha = 90°$. Further increasing α reduces the current until it is fully blocked at $\alpha = 180°$. The reactance of the TCR X_{TCR} (reactor L has a reactance of X_L) is introduced in Eq. (2) [17].

$$X_{TCR} = \frac{\pi X_L}{2(\pi - \alpha) + \sin(2\alpha)} \tag{2}$$

The TSC (the middle component in Fig. 7) has a capacitor connected in series with the thyristors. The thyristors control

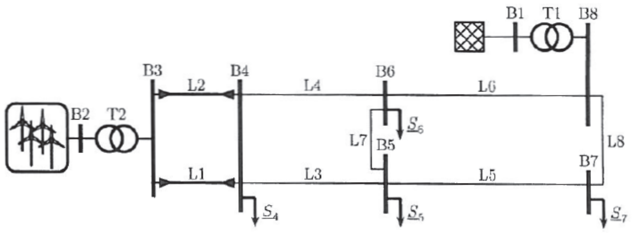

Fig. 8. Grid diagram of a fictitious high voltage grid [17]

the capacitance X_{TSC} in a step-wise manner by switching the capacitor banks on or off [17].

The impedance of the SVC X_{SVC} results from the parallel circuit of the reactor (X_L) and the combined capacitance X_{Cvar} Eq. (3) [17]. The most practical setting is when $X_{Cvar} > X_L$, whereas the practical range of X_{SVC} is limited by the resonance angle [17].

$$X_{SVC} = \frac{\pi X_L}{2(\pi - \alpha) + \sin(2\alpha) + \frac{\pi X_L}{X_{Cvar}}} \tag{3}$$

The input parameters for the SVC device in `pandapower` are X_L, X_{Cvar}, and the voltage set-point. The output values are the inductive or capacitive reactive power of the SVC, and the corresponding thyristor firing angle α.

*2) **Thyristor Controlled Series Compensator**:* Thyristor Controlled Series Compensator (TCSC) is "an impedance compensator which is applied in series on an AC transmission system to provide smooth control of series reactance" [18]. The most common use case of a TCSC device is controlling the active power flow. One of the possible practical implementations is a parallel circuit of a fixed capacitor and a TCR (Fig. 7 [17]). The calculation of the impedance is equivalent to the one of SVC Eq. (3) [17].

*3) **Comparison with an example from literature**:* We replicated the case study of the SVC device from the reference [17] to confirm that our implementation is consistent with the reference. We recreated a model of a fictitious 230 kV high voltage grid with a 500 MW wind park connected via a transformer at bus B3 (Fig. 8). The loads are assumed constant and the wind park feed-in is varied according to a profile.

At first, a benchmark case is obtained via a time-series calculation over 24 hours with time intervals of 15 min (Fig. 9, left side). After connecting an SVC device at bus B7, its voltage is now fixed to the set-point of 230 kV (Fig. 9, right side). The curve shape of the thyristor firing angle matched with the reference, although the exact values could not be reproduced because the specific settings for X_L and X_{Cvar} were not mentioned in the document. These results are consistent with [17], demonstrating that we could successfully reproduce the implementation.

F. Temperature-dependent power flow calculation

Overhead line temperature is the physical limit for defining the line ratings. Including weather data in the power flow calculation allows for considering this limit directly, instead of

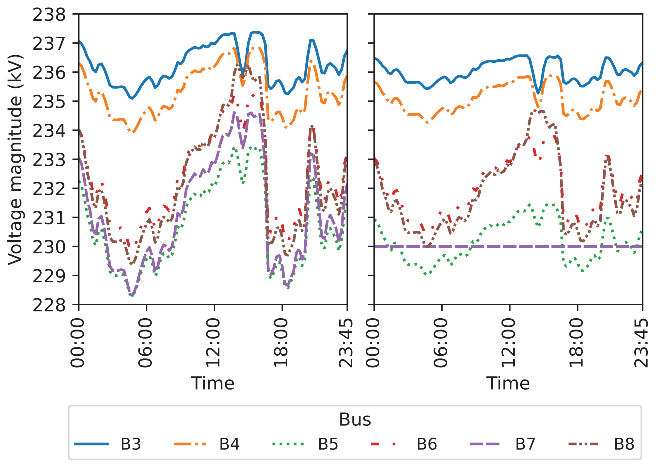

Fig. 9. Voltage variations at the buses for the benchmark case (compare [17])

relying on worst-case assumptions. Including the temperature calculation in a power flow study enables a more accurate consideration of the grid transport capacity. For instance, it is especially useful for analyzing the application of Dynamic Line Rating (DLR). Furthermore, the inclusion of the thermal inertia aspect will be interesting for contingency analysis. We implemented Temperature-Dependent Power Flow (TDPF) calculation in `pandapower` [20] based on the implementations described in [21]–[23].

With the approach of [21] the Jacobian matrix and the mismatch equations are expanded to calculate the line temperature in Newton-Raphson power flow. The thermal resistance of the lines is at the core of this method, because it is used to calculate the temperature-related values in the Jacobian matrix. We followed the method from [22], [23], which is a simplification of the CIGRE model [24], rather than a CIGRE or IEEE standard, because it provides a good balance between performance and accuracy. In addition, we included the approach from [23] to also consider the thermal inertia of overhead lines. We demonstrate the use of this feature, as well as describe the implementation and the required input parameters in more detail in the tutorial[3].

G. Converter from PowerFactory to pandapower

`DIgSILENT PowerFactory` is one of the leading simulation software products in the field of power system study. It is widely used in the industry and in academia. It will be of a great benefit to the `pandapower` community to be able to export the existing grid models from `PowerFactory` to `pandapower`. This opens ways for collaboration between different organizations. Furthermore, it allows to combine workflows that include modeling and analysis with `PowerFactory` and `pandapower`.

We implemented a converter, which allows exporting a grid model from `DIgSILENT PowerFactory` to `pandapower` through a Python interface. The user can either

[3]https://github.com/e2nIEE/pandapower/blob/develop/tutorials/temperature_dependent_power_flow.ipynb

set up a "user-defined tool" in `PowerFactory`, or use a Command Line Interface (CLI). The "user-defined tool" allows starting the converter from within `PowerFactory` and using a Graphic User Interface (GUI) to specify some of the options of the converter, as well as set the path for saving the `JSON` file of the grid in `pandapower` format. The CLI can be used as part of a Python script e.g., to automate exporting of a large number of grid models. The converter implementation provides automated testing of the exported grid by comparing the results of a power flow calculation in `PowerFactory` and `pandapower`. A `PowerFactory` instance must be installed and the license must be available for the converter to work, either in the variant of GUI or CLI.

The documentation[4] describes the configuration and usage of the converter, and a tutorial[5] demonstrates the use of the converter via CLI with an example grid. The documentation also lists the supported PowerFactory components.

H. *Converter from CIM CGMES to pandapower*

A second converter that was recently developed is `cim2pp`, which translates CIM CGMES version 2.4.15 XML files into the `pandapower` format. It is written in Python, validated with CGMES test networks and since December 2022 it can be downloaded as part of the open source `pandapower` library. A technical documentation[6] explains the basic usage and contains an Ipython-tutorial for new users.

The source input for `cim2pp` is CGMES grid data, which can either be stored as a list of single XML files or ZIP file(s). In the case of a ZIP archive, it will be recursively searched for XML files, including subfolders of any depth. We have implemented this feature because some software packages with CIM export functionality will generate ZIP archives in multiple ways. Moreover, it can occur that CGMES-structures are used for other purposes and are enriched with additional data. For these reasons, a built-in parser searches the archive only for CGMES XML files and ignores any other files.

The Python standard library is then employed to generate an XML tree for each imported CGMES file. The data in these trees are processed further - they contain namespaces, which must be interpreted accordingly. The translated data are stored in a middle layer structure, similar to that of `pandapower`, which consists of dictionaries and DataFrames. In a final step, this middle layer is mapped into a `pandapower` network. Apart from the usual column entries, `cim2pp` will generate some additional data columns through which one can refer back to the original CGMES file for convenience. For example, the `net.bus` table will get a new column `origin_id` which contains the CIM ID of each corresponding bus.

III. Open Source Software `pandapipes`

Current and future energy networks are not limited to electricity grids. Especially in the light of the upcoming decarbonization challenges in various sectors, the importance of pipe networks (e.g. for district heating and hydrogen) continues to grow. Against this background, `pandapipes` has been developed as a separate but closely related tool to `pandapower` for the simulation of pipe networks [3]. The recent developments of `pandapipes` have been focusing on applicability to large-scale, real-world gas and heating grids. For this, accelerated internal calculations, new components and an import function for `STANET`-grid models have been developed.

A. *Speed-up of internal calculations*

The computational performance of `pandapipes` has been improved by re-organizing the derivatives and by partial code compilation. For the former, the derivatives that comprise the system matrix are no longer calculated in the different branch components but globally, since they all follow the general friction model in a pipeline. For components where this behavior is not desired, including some of the new components described in Section III-B, the equations can be changed within the component models after the global formulation. For the latter, a *Numba*-based just-in-time (jit) compilation [25] has been implemented for the calculation of derivatives for the friction head losses of compressible and incompressible media, the Darcy friction factor (Nikuradse and Colebrook-White model) and some auxiliary functions as the intermediate pressure in pipes for property calculation of the flowing medium. The user can choose between using the jit-compiled functions and the previous NumPy-based implementations. The usage of jit-compiled functions is mainly useful in the case of repeated simulations, as the compilation is lengthy and always takes place upon the first run.

B. *New component types*

`pandapipes` has been extended by three new component types: pressure regulator, flow control, and compressor. All three have an inlet junction i and outlet junction j. In this initial implementation, possible temperature changes are neglected.

The pressure regulator applies a pressure lift/drop Δp_{ij} between i and j such that a pre-defined pressure at a control junction k is reached ($k = j$ is possible). It is meant to be an ideal representation of a pressure regulator station that connects the gas transmission and distribution level, but also allows for pressure increases. The user can also define the junction at which the pressure ought to be fixed. If this junction does not correspond to the outlet junction, the pressure regulator serves as a wide-area control model.

The flow control component enforces a pre-defined mass flow $\dot{m}_{ij}$ between the inlet and outlet junctions i and j. It can be seen as an ideal representation of a partially opened valve or a component with a specific head loss coefficient that could be controlled from outside. This behavior is especially useful for the simulation of district heating grids, where the flow and return are normally connected directly via heat exchangers, representing household heating systems. In real grids, the flow through these heat exchangers is controlled based on the required heat of the household, which would be too complex to

[4]https://pandapower.readthedocs.io/en/latest/converter/powerfactory.html
[5]https://github.com/e2nIEE/pandapower/blob/develop/tutorials/converter_powerfactory.ipynb
[6]https://pandapower.readthedocs.io/en/latest/converter/cgmes.html

model in a large district heating grid model. The flow control component can therefore set the required mass flow through the heat exchanger, which in many scenarios is known at the time of the simulation.

The simplified compressor model uses a relative pressure increase factor (*compression ratio* Π, referring to absolute pressures) that is an input parameter of the element. For reverse flow, bypassing is assumed. For a compressor with a mass flow m_{ij}, the outlet pressure is calculated as follows:

$$p_j = \begin{cases} p_i \cdot \Pi, & \text{if } \dot{m}_{ij} > 0\,\text{kg/s} \\ p_i, & \text{otherwise} \end{cases} \tag{4}$$

To represent the advanced operation of the compressor, Π may be set as a dependent variable by adding a *basic controller* that is based on a state variable (e.g., the flow at the compressor's inlet) in the net.

The required ideal compression power $P_{\text{compr},ij}$ in Megawatt for compressors with $\dot{m}_{ij} > 0$ is approximated with the adiabatic change in enthalpy by applying Eq. (5) [26].

$$P_{\text{compr},ij} = \dot{m}_{ij} \frac{\kappa}{\kappa - 1} R_\text{s} z(p_i) T_i \cdot \left(\Pi^{\frac{\kappa-1}{\kappa}} - 1 \right) \cdot 10^{-6} \tag{5}$$

With isentropic exponent κ (assumed as 1.4), specific gas constant R_s, compressibility $z(p)$ and temperature T_i.

C. Converter from `STANET` *to* `pandapipes`

`STANET`[7] is one of the leading proprietary software products for gas grid analysis in Germany. In order to make `STANET` grid models easily usable in `pandapipes`, we extended the converter module of `pandapipes`, enabling the user to read a grid model file (*.csv*-format) exported from `STANET` and convert the `STANET` database tables into `pandapipes` tables. The converter is an ongoing project and currently covers the conversion of gas grid models, that has been tested on a small number of realistic grid models. As district heating networks are modeled in a different way in `STANET`, the converter will be adapted in future to derive executable grid models.

One of the main differences in gas grid modeling between `STANET` and `pandapipes` is the usage of household connection systems. In `STANET`, all pipes and nodes that belong to this system are stored in separate database tables. During the conversion, the pipe and household pipe tables are integrated and just tagged in `pandapipes` in order to differentiate between them. Also, the sinks that usually belong to the household system are presented with the two alternative nodes that `STANET` uses internally to simulate or neglect the household system. This way, the user can restore the two `STANET` simulation modes.

In `STANET`, valves are modeled with a small length, which is not the case in `pandapipes`, where valves always have a length of 0. Therefore, in the `pandapipes` model the valves are converted into two components by default, a valve and a pipe. This grants coherence with the default grid design while retaining a similar behavior as in `STANET`. For even higher precision and better comparability, a new *valve_pipe* component type has been integrated into the converter for increased compatibility. However, this component model is not intended as a core component of `pandapipes`.

[7]Fischer-Uhrig Engineering, STANET, https://www.stafu.de/en/home.html

IV. Conclusions and Outlook

Both `pandapower` and `pandapipes` are in active development with growing user and developer communities.

With the development and extension of the new short-circuit calculation with current and voltage angles, we are developing directional overcurrent relay and distance relay. For both these relays, we need the complex voltage and current phasors as input parameters, which will be accomplished with the implementation of new short-circuit calculation. In addition to the implemented FACTS devices, more can be added in future work, for example HVDC converters. The implemented TDPF will enable more detailed power system analysis, especially in conjunction with contingency analysis.

For `pandapipes`, new components and an import function were implemented to widen the range of possible applications. Ongoing and future developments of `pandapipes` include mixtures of gases, transient simulation, especially for district heating networks, and improved visualization functions.

The addition of more converters to `pandapower` and `pandapipes` simplifies the workflows for the analysis of electric and gas grids for the community, reduces barriers to collaboration, and enables integrating different software products in the same workflow.

Acknowledgments

This work presents selected developments within the projects "RESIST" (Grant No. 03SF0637), "RobustPlan" (Grant No. 03EI6037B), and "ANaPlan Plus" (Grant No. 03EI4032A), funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK, formerly known as the German Federal Ministry for Economic Affairs and Energy), based on a decision of the Parliament of the Federal Republic of Germany. We thank 50Hertz Transmission GmbH for funding the developments of the short-circuit calculation. The contents of this publication are the sole responsibility of the authors. We thank Ara Panosyan, PhD, and Bonface Ngoko, PhD, for their support by answering our questions.

References

[1] W. McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, edited by Stéfan van der Walt and Jarrod Millman, Eds., 2010, pp. 56 – 61.

[2] L. Thurner, A. Scheidler, F. Schäfer, J.-H. Menke, J. Dollichon, F. Meier, S. Meinecke, and M. Braun, "pandapower - an open source python tool for convenient modeling, analysis and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6510–6521, November 2018.

[3] D. Lohmeier, D. Cronbach, S. R. Drauz, M. Braun, and T. M. Kneiske, "Pandapipes: An open-source piping grid calculation package for multi-energy grid simulations," *Sustainability*, vol. 12, no. 23, 2020. [Online]. Available: https://www.mdpi.com/2071-1050/12/23/9899

[4] R. Bolgaryn, G. Banerjee, D. Cronbach, S. Drauz, Z. Liu, M. Majidi, H. Maschke, Z. Wang, and L. Thurner, "Recent developments in open source simulation software pandapower and pandapipes," in *2022 Open Source Modelling and Simulation of Energy Systems (OSMSES)*, 2022, pp. 1–7.

[5] W. Schossig and T. Schossig, *Netzschutztechnik*, 7th ed., ser. Anlagentechnik für elektrische Verteilungsnetze. Berlin: VDE Verlag GmbH, 2021.

[6] VDE e.V., "Schutz- und Automatisierungstechnik in aktiven Verteilnetzen." [Online]. Available: https://www.vde.com/de/etg/publikationen/studien/studie-sua

[7] U. Spindler, *Schutz bei Überlast und Kurzschluss in elektrischen Anlagen: Erläuterungen zur neuen DIN VDE 0100-430:2010-10 und DIN VDE 0298-4:2003-08*, 3rd ed., ser. VDE-Schriftenreihe - Normen verständlich. Berlin: VDE-Verl., 2010, vol. 143.

[8] ALSTOM, *Network protection and automation guide: Protective relays, measurement and control*. Great Britain: Alstom Grid, 2011.

[9] P. Lytaev, G. Banerjee, and J. Haack, "Adaptive and automated protection settings for over-current relays in radial grid configuration," in *PESS 2021; Power and Energy Student Summit*, 2021, pp. 1–6.

[10] I. Kasikci, *Short circuits in power systems: a practical guide to IEC 60909-0*. John Wiley & Sons, 2018.

[11] International Electrotechnical Commission, *Measuring relays and protection equipment - Part 1: Common requirements*, ser. International standard Norme internationale. Geneva: International Electrotectnical Commission, 2009, vol. IEC 60255-1.

[12] Z. Liu, J.-H. Menke, N. Bornhorst, and M. Braun, "Static grid equivalent models based on artificial neural networks," *IEEE Access*, vol. 9, pp. 168 535–168 546, 2021.

[13] T. E. Dy Liacco, S. C. Savulescu, and K. A. Ramarao, "An on-line topological equivalent of a power system," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-97, no. 5, pp. 1550–1563, Sep. 1978.

[14] J. B. Ward, "Equivalent circuits for power-flow studies," *Transactions of the American Institute of Electrical Engineers*, vol. 68, no. 1, pp. 373–382, July 1949.

[15] F. Aschmoneit and J. Verstege, "An external system equivalent for on-line steady-state generator outage simulation," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-98, no. 3, pp. 770–779, 1979.

[16] P. Dimo, *Nodal analysis of power systems*, ser. Abacus Bks. Editura Academiei Republicii Socialisté România, 1975. [Online]. Available: https://books.google.de/books?id=4dAiAAAAMAAJ

[17] A. Panosyan, "Modeling of advanced power transmission system controllers," Ph.D. dissertation, Gottfried Wilhelm Leibniz Universität Hannover, 2010.

[18] A.-A. Edris, R. Adapa, M. H. Baker, L. Bohmann, K. Clark, K. Habashi, L. Gyugyi, J. Lemay, A. S. Mehraban, A. K. Myers, J. Reeve, F. Sener, D. R. Torgerson, and R. R. Wood, "Proposed terms and definitions for flexible AC transmission system (FACTS)," *IEEE Transactions on Power Delivery*, vol. 12, no. 4, pp. 1848–1853, 1997.

[19] Circuit Diagram. (2022) Circuit diagram web editor. [Online]. Available: https://www.circuit-diagram.org/editor/

[20] R. Bolgaryn, J. Wiemer, A. Scheidler, and M. Braun, "Security-constrained active power curtailment considering line temperature and thermal inertia," *IET Generation, Transmission & Distribution*, 2022, in review.

[21] S. Frank, J. Sexauer, and S. Mohagheghi, "Temperature-dependent power flow," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4007–4018, 2013.

[22] B. Ngoko, H. Sugihara, and T. Funaki, "Validation of a simplified model for estimating overhead conductor temperatures under dynamic line ratings — comparison with the CIGRE model," *IEEJ Transactions on Power and Energy*, vol. 138, no. 4, pp. 284–296, 2018.

[23] ——, "A temperature dependent power flow model considering overhead transmission line conductor thermal inertia characteristics," in *2019 IEEE International Conference on Environment and Electrical Engineering and 2019 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. IEEE, 2019, pp. 1–6.

[24] CIGRE Working Group 22.12, "Thermal behaviour of overhead conductors," Aug. 2002.

[25] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A LLVM-based Python JIT compiler," in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, ser. LLVM '15. New York, NY, USA: Association for Computing Machinery, 2015. [Online]. Available: https://doi.org/10.1145/2833157.2833162

[26] M. Schmidt, M. C. Steinbach, and B. M. Willert, "High detail stationary optimization models for gas networks," *Optimization and Engineering*, vol. 16, no. 1, pp. 131–164, 2015.