A Seamless Workflow from Open-Source Planning Models to Industry-Standard Stability Simulations

1st Daniel Jung
DLR-Institute of Networked Energy Systems
German Aerospace Center
Oldenburg, Germany
daniel.jung@dlr.de

2nd Saikrishna Vallabhaneni
DLR-Institute of Networked Energy Systems
German Aerospace Center
Oldenburg, Germany
saikrishna.vallabhaneni@dlr.de

3rd Frank Schuldt
DLR-Institute of Networked Energy Systems
German Aerospace Center
Oldenburg, Germany
frank.schuldt@dlr.de

4th Karsten von Maydell
DLR-Institute of Networked Energy Systems
German Aerospace Center
Oldenburg, Germany
karsten.maydell@dlr.de

***Abstract*—The ongoing transition toward decentralized generation and inverter-based resources has heightened the need for robust simulation methodologies capable of evaluating system stability under a wide range of operating conditions. Nevertheless, restricted access to real world grid data and prevailing confidentiality constraints hinder the dissemination of detailed models, thereby limiting collaborative research and reproducibility. Although open frameworks such as Python for Power System Analysis (PyPSA) offer transparent and flexible tools for system optimization and scenario development, they currently lack advanced analytical capabilities, particularly for dynamic simulation. To address this deficiency, we present pypsa2pf, a Python-based converter that translates PyPSA network representations into DIgSILENT PowerFactory projects. The converter automatically maps network topology, equipment specifications, control schemes, and time-series data to PowerFactory, requiring minimal subsequent manual intervention. The utility of pypsa2pf is illustrated through its application to a multi-sectoral 2045 reference network for Northwest Germany. Validation of the conversion is performed by comparing direct-current (DC) load-flow results of the PowerFactory model with those obtained from the original PyPSA model. The resulting model reproduces the original network parameters and operating set points with high fidelity, requiring only minimal adjustments to enable AC load-flow and dynamic power-system analyses. pypsa2pf markedly streamlines the workflow from open-source scenario development with PyPSA to industry-grade stability assessment in PowerFactory, facilitating more efficient and collaborative power system studies.**

***Index Terms*—PyPSA, DIgSILENT PowerFactory, open source, decentralized generation, HVDC, AC load flow, dynamic simulation, stability analysis**

I. Introduction

The increasing penetration of renewable energy generators and inverter-based resources fundamentally reshapes the way electric networks are planned and operated. Simulations still play a crucial role in this shift by enabling operators to evaluate system behavior under a wide range of conditions, ensuring system stability and resilience through both static and dynamic studies on realistic grid models. However, access to detailed network data from grid operators is often restricted or incomplete, potentially delaying studies. In addition, confidentiality restrictions frequently prevent publication of real grid models, limiting collaboration and reproducibility. For example, Python for Power System Analysis (PyPSA) [1] enables transparent scenario development and steady-state optimization, but it lacks high-fidelity AC load flow, detailed component and control modeling, fault analysis, and time-domain dynamic simulation capabilities required for advanced stability studies. Such features are typically only available in commercial environments such as DIgSILENT PowerFactory, which is an industry-proven tool for static and dynamic simulations, including many useful analysis functions for studying electrical transmission and distribution grids.

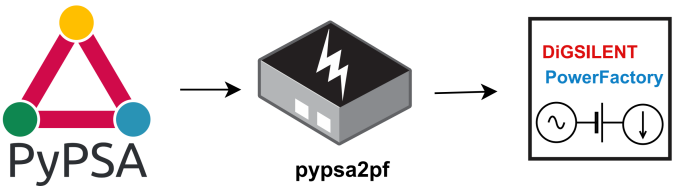

Fig. 1. Illustration of the pypsa2pf workflow. PyPSA logo used with permission © PyPSA contributors.

Several recent studies have addressed the challenges and uses of translating power system models between open-source and commercial simulation platforms. Nika et al. [2] have demonstrated the importance of linking PyPSA-based expansion planning with PowerFactory dynamic simulations for the integration of large-scale electrolyzers, but they have not provided further details on the used Python scripts doing the conversion, nor have they clarified their intentions to make the toolbox open-source. Also Rios-Peñaloza et al. [3] have coupled PyPSA to PowerFactory to perform dynamic stability analyses on scenarios developed on the

basis of PyPSA-Eur [4], but the Python-based tool facilitating the coupling is not further described either. In addition, there are Python packages providing an improved (easier-to-use) Python-PowerFactory interface, such as PowerFactory-Tools [5] or powfacpy [6]. The former also offers the export of PowerFactory network data to the Power System Data Model format for external analysis. Also, PandaPower includes code to import grid data from PowerFactory [7]. But to the knowledge of the authors, there is no publicly available tool to transfer grid models from one of established Python-based open-source modeling environments (e.g. PandaPower, PyPSA) to PowerFactory.

To address this gap, the Python-based converter pypsa2pf was developed to automate the transfer of power grid models from PyPSA to DIgSILENT PowerFactory (see. Fig. 1). Similar to the tools described in the mentioned literature [2] [3], the converter maps the network topology, equipment data, control modes, and load and generation time series into an operational PowerFactory project, with little need for additional manual adjustments. Sector-coupling components are represented by equivalent generators and loads, and high-voltage direct current (HVDC) links are configured with distinct operational modes for offshore wind connections as well as onshore interconnectors to ensure realistic operational behavior. The converter is intended to be published as open-source software to enable reproducible workflows. All modeling assumptions will be transparently documented.

In this study, we demonstrate the capabilities of pypsa2pf by applying it to a model of the electricity system of Northwest Germany in the year 2045, which is based on the 2035 eGo^n model [8]. The grid surrounding the region of interest is simplified by an extended PyPSA clustering algorithm. With only a few additional manual steps, the converted PowerFactory model can be made suitable for AC load flow analysis in PowerFactory. The recommended steps to prepare an open-data-based grid model for dynamic analysis are currently under development.

This study introduces the main concepts of pypsa2pf and demonstrates the accurate mapping of the PyPSA model to PowerFactory. It focuses on the development and verification of the converter and the overall workflow. The paper is structured as follows: After Section I has introduced pypsa2pf and its objectives, Section II presents the pypsa2pf methodology and introduces the power system model used for demonstration and the post-processing steps performed manually in PowerFactory to enable AC load flow analysis. Section III compares the PowerFactory model with the PyPSA model to showcase pypsa2pf's validity. Finally, Section IV concludes the paper and outlines suggested further development.

II. Methodology

This section briefly introduces the methodological framework underlying pypsa2pf and the test models used within this study. The following subsections detail the implementation of pypsa2pf, transformation procedures, and the grid model used for demonstration.

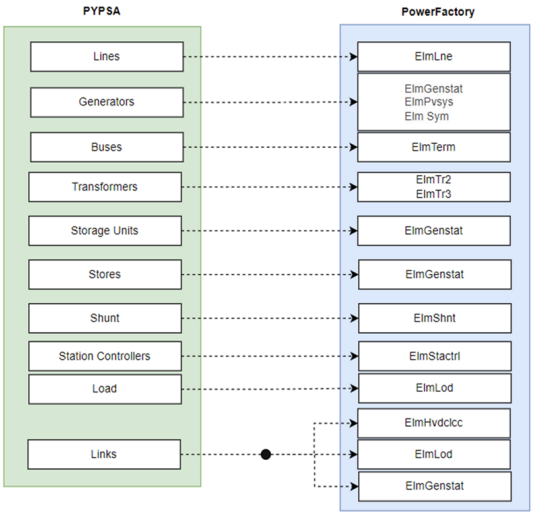

Fig. 2. Mapping of PyPSA power grid components to corresponding PowerFactory types.

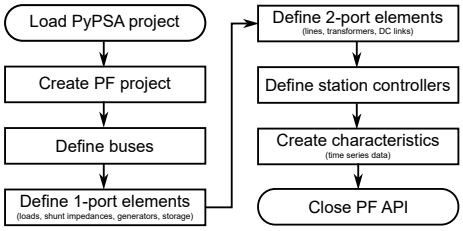

Fig. 3. Main processing steps in the PyPSA-to-PowerFactory converter.

A. The PyPSA-PowerFactory Converter (pypsa2pf)

A Python module for the automatic conversion of PyPSA grids into PowerFactory models was developed. The implementation, referred to as "pypsa2pf", will be part of the Python package "pypfconvert", which is intended to become a toolbox for translating power system models from multiple open-source grid formats into PowerFactory. The package is planned for release under an open-source license, with pypsa2pf as one of its conversion modules.

With the help of pypsa2pf, it is possible to generate PowerFactory models from publicly available grid data, enabling dynamic simulations and other types of analysis that are typically not possible with existing open-source tools. It can convert power system models from PyPSA that include features such as distributed generators, storage systems, fuel cells, and HVDC systems. pypsa2pf uses a predefined mapping to translate PyPSA object types into their corresponding counterparts in PowerFactory (see Fig. 2). The converter is intended to produce a faithful representation of the original model, making as few assumptions as possible to conform to PowerFactory standards. In the following, we highlight some key processing steps performed by pypsa2pf (see Fig. 3), as well as some manual post-processing steps.

Representation of Generators, Loads and Storage Units: pypsa2pf assigns loads, generators, storage units and other

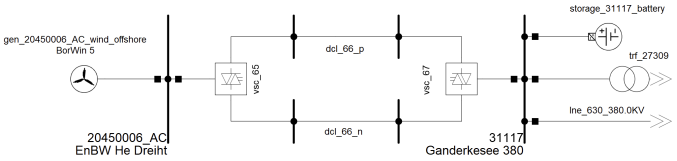

Fig. 4. Typical pypsa2pf representation of an HVDC system with voltage-source converter technology (here: BorWin 5).

components to their corresponding DIgSILENT PowerFactory element types. Loads are represented as static PQ elements with active and reactive power profiles transferred directly from existing PyPSA optimization results. Generators are represented by synchronous or static generator models, depending on the type of technology. Storage units are implemented as static generators with finite positive and negative active power limits. This representation maintains electrical equivalence between both platforms while enabling subsequent static and dynamic analyses in PowerFactory.

Representation of HVDC Infrastructure: Several HVDC systems are expected to enter into service in the coming years, partially to connect offshore wind farms to the onshore grid, but also as additional means to transport large quantities of electrical energy over vast distances, both onshore and offshore. The involved inverter systems could also play a pivotal role in maintaining grid stability in the future, being able to provide various system services. For this reason, we opt for modeling HVDC systems explicitly in the PowerFactory grid model, using available component models.

In PyPSA, HVDC connections are represented as links, a versatile component type that can be used for any form of energy transport or conversion with a fixed efficiency factor. In PowerFactory, links representing DC connections are converted either to a pair of voltage-source converters (VSCs) with two DC line elements (positive and negative poles) and four DC nodes (see Fig. 4), or alternatively to a single object representing a DC link with line-commutated converter (LCC) technology. Converters are properly configured, i.e., in the case of single VSC objects, it is made sure that one of the converters controls the DC voltage. Custom control options can be set up a posteriori.

Active Power Control: By default, pypsa2pf considers the generator control mode specified in the PyPSA project. Optionally, the control mode of synchronous machines can be overwritten and set to "const V" mode (controlling the voltage). If there are multiple generators (or converters) connected to a single bus that control the voltage, pypsa2pf can automatically create a station controller in PowerFactory that handles the provision of reactive power on that bus.

B. Energy System Model of Northwest Germany in 2045

To demonstrate the pypsa2pf workflow, we apply it to a model recently developed in a research project in which the eGo^n model [8], which features a scenario of the German energy system in 2035 (eGon2035), was upgraded to match the future 2045 German energy system. The upgrade includes

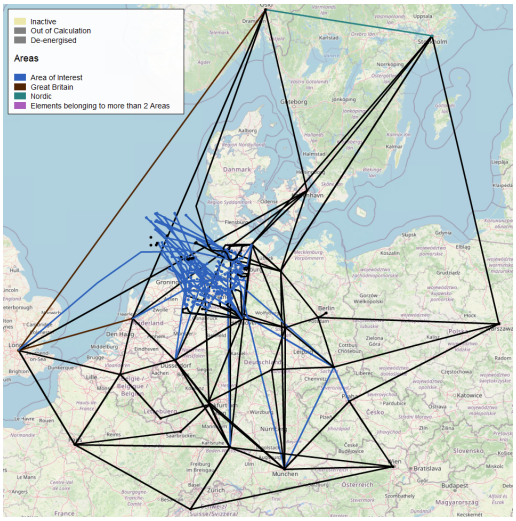

Fig. 5. Resulting PowerFactory grid model, showing region of interest (blue) and simplified (clustered) surroundings (green/black). Background © OpenStreetMap contributors.

new and reinforced AC and DC infrastructure based on the 2037/2045 network development plan from 2025 [9] and adjusted load and generation time series based on the latest long-term scenarios (Langfristszenarien [10] [11]) for 2045 which were developed on behalf of the German Federal Ministry of Economic Affairs and Energy (BMWE). They outline the long-term grid expansion plans for Germany in several scenarios. Among these scenarios, the O45-Strom scenario was selected, which considers an increased level of electrification. The projected renewable generation and demand data per federal state were extracted and mapped onto the nodes of the eGo^n grid model.

The electricity infrastructure upgrades based on the latest network development plan [9] covered lines, buses, transformers and HVDC links. Components already considered within the eGo^n model are identified and excluded from the upgrade. The project also covered modeling other sectors, including a model of the planned hydrogen core grid (German: Wasserstoff-Kernnetz [12] [13]). Coupling points with this part of the multi-sector PyPSA model are represented as loads and generators with respective time series in PowerFactory.

An "area of interest" (Ems-Elbe area) was defined within the model in which full grid details are maintained, while the grid outside of this area was simplified (see Fig. 5) . To that end, the clustering algorithm included in PyPSA was extended to keep the distinction between different voltage levels intact, as well as to correctly handle point-to-point HVDC infrastructure.

A linear optimization algorithm of the sector-coupled model in PyPSA was performed using the Gurobi solver [14], minimizing total system costs considering all hours of the year 2045 (8760 hours). The optimization problem also included an expansion problem to decide over the built-out capacity of certain components, e.g. of electrolyzers and fuel cells. The optimization results are used as hourly unit commitment in PowerFactory.

C. Manual Post-Processing Steps in PowerFactory

System optimization models such as PyPSA typically only consider DC load flow (linear approximation). Hence, voltage, reactive power, and line losses are omitted. To make the model fit for AC load flow and make full use of PowerFactory's user interface capabilities, it is pertinent to take some additional post-processing steps directly in PowerFactory (see Fig. 6). While some steps could be automated in a future version of pypsa2pf, some custom settings are likely to remain necessary, depending on the user's needs and the study to be performed.

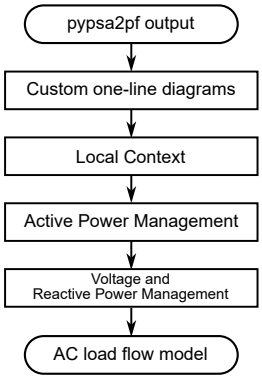

Fig. 6. Suggested post-processing steps to be done manually in PowerFactory.

Custom One-line Diagrams: Initially, the PowerFactory model only contains the standard line diagram showing a topological view of the whole system. A geographic diagram can easily be added. While the topological diagram can be quite confusing, featuring overlapping components, it also does not show the elements in their natural geographic location. To properly prepare and study the grid model, it is helpful to define custom-made one-line diagrams, each showing only a part of the system. As an example of a custom one-line diagram, Fig. 7 shows a part of the 110 kV grid near Emden. These kind of grid visualizations greatly help with model preparation and scenario analysis, providing a level of understanding of the power system model that is hardly possible with common open-source tools.

Local Context: As in many other models, the eGo^n grid uses arbitrary labels (random numbers) for most grid components. To improve the usability of the model, it is helpful to add real-world names to certain components. We suggest to use PowerFactory's chr_name parameter to store the real-world names. This way, the original labels (loc_name) of the components remain unchanged so that the relationship with their PyPSA counterparts is maintained. Both parameters can be shown side-by-side within diagrams (see Fig. 7).

Active Power Management: While optimization models such as PyPSA do not require a slack bus for balancing, it is required to set up a slack bus for performing AC load flow analyses. However, a single slack bus would not be realistic for a large integrated transmission grid such as the one used in this study for demonstration purposes. In the real grid, load and generation are balanced by a large

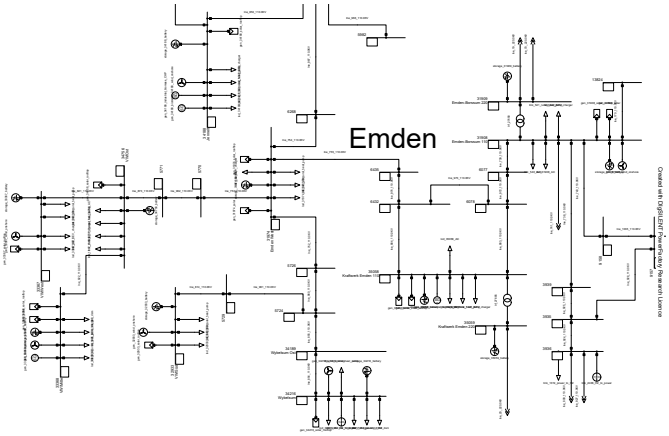

Fig. 7. Schematic diagram of the modeled 110 kV grid near Emden.

TABLE I
ASSIGNMENT OF FCR CAPACITY TO THE NODES EXISTING IN THE GRID MODEL.

Node	Representing FCR of...	FCR (MW)	Share (%)
FR	FR + ES + PT	864	27.7 %
AT	AT + HR, SI, BA, ME/RS, GR, AL, MK, XK, MD, BG, TR	612	19.6 %
DE (total)	DE (all federal states) + LU	564	18.1 %
CH	CH + IT	372	11.9 %
PL	PL + UA	276	8.9 %
CZ	CZ + SK + HU + RO	201	6.5 %
NL	NL	110	3.5 %
BE	BE (LU integrated)	93	3.0 %
DK	DK (CE-part)	25	0.8 %
Sum		3 117	100 %

number of assets distributed over all countries within the Continental European Synchronous Area. We therefore set up a distributed slack spanning all countries and all German federal states, with weighting factors aligned to the planned frequency containment reserve (FCR) capacities. As we cannot know future planned FCR capacities, we make use of data from ENTSO-E's latest Load-Frequency Control (LFC) Annual Report [15] which provides data for the initially obliged FCR capacity per Member State. For simplicity reasons, only one asset (sufficiently large generator or battery system) is chosen per node to provide balancing power.

Although our model includes only Germany and its directly connected neighboring countries, it is pertinent to reflect the full FCR capacity of the Continental European Synchronous Grid for balancing. To that end, the FCR contribution of other countries was assigned to one of the nodes existing in the model (heuristic approach). Example: France is assigned the combined FCR capacity of France, Spain and Portugal. Table I reports the chosen FCR distribution among the model's nodes representing European countries.

We also model nodes outside the Continental European

Synchronous Area (United Kingdom, Sweden, Norway). For that reason, the total FCR is larger than the commonly known value of 3 GW for the Continental European grid. The assumed FCR contribution of Finland is added to that of Sweden.

Voltage and Reactive Power Management: To balance the system in terms of reactive power, we first configure all load elements that represent MV/HV substations with a constant power factor of 0.98 inductive (typical value for mixed residential/commercial consumption). The loads of other categories (e.g. electrolyzers, heat pumps, EV charging) are left to $\cos(\phi) = 1$.

Next, we choose that only synchronous machines and VSCs control voltage, while static generators are switched to "const Q" mode. As many others, this choice depends solely on the requirements and modeling assumptions of the study performed. In cases where small generators would provide an excess of reactive power, we switch them to "const Q" mode as well and set a constant power factor (although we keep $\cos(\phi) = 1$ in most cases). Furthermore, for all synchronous machines, we consider PowerFactory's standard capability curve.

pypsa2pf already creates station controllers for all buses with multiple components controlling the voltage. It is considered sensible to switch all station controllers' reactive power distribution to the mode "According to Rated Power" (parameter imode = 1) to obtain a better distribution of reactive power contribution among participating machines. For all generators (except for slack generators), the option "Out of service when active power is zero" is activated, preventing non-dispatched generators (where $P = 0$) from providing reactive power.

In the considered grid model, we initially observe reactive power losses of ca. 10 Mvar. To further optimize the system, we suggest to take the following steps (as necessary):

1) Adjust target voltages of generators and station controllers (we initially set them to 1.05 p.u. and only changed them slightly in case of constraint violations);
2) Use all available assets to provide reactive power, including PV systems and static generators, setting either a power factor different from 1 (something between -0.95 p.u, and +0.95 p.u., in line with German grid code [16], [17]) or setting up Q(V) curves;
3) Install reactive power compensation devices as needed: shunt capacitors, shunt inductors, or static synchronous compensators (STATCOMs).

In the considered grid model for Northwest Germany (and for the chosen hour), we were able to omit most of the above steps, as the existing assets (synchronous generators, VSCs) were able to provide sufficient reactive power. Slight adjustment of some voltage set points was enough to avoid voltage-related constraint violations that appeared in certain N-1 contingency cases.

For 220 kV and 380 kV, we use voltage constraints of [0.95 p.u., 1.25 p.u.], while for 110 kV, we use [0.95 p.u., 1.15 p.u.]. We have moderately increased the voltage set points to 1.05 p.u. to avoid minimum voltage constraint violations in certain N-1 contingency cases.

TABLE II
TOP FIVE SLACK GENERATORS IN TERMS OF ACTIVE POWER MISMATCH P_{MISM} IN THE HOUR DECEMBER 22, 2045, 9:00 A.M. (HOY=8529).

Generator	P_{PyPSA} [MW]	P_{PF} [MW]	P_{mism} [MW]
13153 nuclear	6.573	6.538	0.0348
13975 industrial biomass chp	0.512	0.509	0.0029
43788 nuclear	7.591	7.588	0.0025
15350 industrial biomass chp	0.00	-0.002	0.0023
13225 industrial biomass chp	0.480	0.477	0.0023

III. VALIDATION RESULTS

To validate the PowerFactory model generated by pypsa2pf, we first check if all components and data were successfully transferred. This verification step comprises a quantitative comparison of component counts for each category, coupled with a visual inspection of the generated topology to confirm its correctness. Using a Python script, we also check that all time series ("time characteristics" in PowerFactory) attached to the various components agree with their PyPSA counterparts. The comparison shows that the time series agree to within machine-numerical precision (differences on the order of 10^{-15}). This already shows that the data has generally been transferred correctly.

In a second step, we perform a DC load flow in PowerFactory (for selected hours) and compare the resulting active power output of generators in PowerFactory with the active power set points (unit commitment) received from PyPSA, resulting in the active power mismatch of generator i,

$$P_{\text{mism},i} = P_{\text{PyPSA},i} - P_{\text{PF},i} \quad . \tag{1}$$

Note that only the power output of slack generators is variable.

The result is that also the DC load flow results agree to a high degree with the results of the PyPSA optimization algorithm which makes use of the linearized load flow equations (DC approximation) as well. The results show that the active power mismatch $P_{\text{mism},i}$ at the slack generators is very small though not completely zero (see Table III). Using DC load flow, we observe a total mismatch of only 54.5 kilowatts which can be explained by the tolerance level used for the PyPSA optimization. Note that this is only a fraction of the total load/generation in that particular hour (ca. 433 GW).

IV. CONCLUSIONS

This work introduced the new Python-based tool pypsa2pf that is able to transfer PyPSA power networks to the proprietary power system analysis software DIgSILENT PowerFactory. The tool was validated by comparing DC load flow results using a recently developed multi-sector energy system model of Northwest Germany. Following the transfer, additional post-processing steps within PowerFactory were suggested to prepare the model for AC load flow calculations. It was demonstrated how a structured conversion workflow

between energy system optimization in PyPSA and detailed simulation in DIgSILENT PowerFactory substantially reduces the time required to move from scenario development to power system stability assessment.

The conversion process also provided insights into the problems of transforming open-data grid models designed for linear optimization using DC approximations into functional AC load flow models. With pypsa2pf, a functioning PowerFactory model can be generated with limited manual adjustments. However, the validation process revealed shortcomings in publicly available datasets, such as eGo^n, particularly regarding the grid topology and parameter accuracy at the 110 kV level. These limitations indicate that publicly available network data must be carefully reviewed and refined before attempting steady-state AC load flow or dynamic simulations.

At its current stage, the converted model is capable of performing AC load flow analyses. Dynamic simulations have not yet been attempted, as they require setting up dynamic models and properly defined initial conditions for each grid component, which is not yet automated and needs to be carefully reviewed. Also, as PyPSA features AC load flow calculations as well, the validation process should be extended to AC load flow. Addressing these issues is part of ongoing work aimed at extending the functionality of the tool to support the conversion of any PyPSA network to run dynamic simulations in DIgSILENT PowerFactory. Another field of activity is setting up a co-simulation approach between PowerFactory and a hydraulic gas grid model (possibly PandaPipes [18] [19] [20]) based on the same multi-sector PyPSA energy system model, which will make it possible to study resilience scenarios under consideration of gas-power interdependencies.

Acknowledgements

We kindly thank our former colleagues Ingo Liere-Netheler and Elena Memmel for the development of the initial version of the pypsa2pf converter. We also thank our colleagues Oriol Raventos Morera, Lars Höpken, Rita Kastrati, Ontje Lünsdorf, Bruno Schyska and Kateryna Volochay for the fruitful collaboration in creating the multi-sector energy system model of Northwest Germany in PyPSA and developing the accompanying tools.

References

[1] T. Brown, J. Hörsch, and D. Schlachtberger, "PyPSA: Python for Power System Analysis," Journal of Open Research Software, vol. 6(1), 2018, DOI:10.5334/jors.188.

[2] E. Nika et al., "Large-Scale Integration of Electrolyzers into the German Transmission System: A 2030 Scenario," 2025 IEEE Kiel PowerTech, Kiel, Germany, 2025, DOI:10.1109/PowerTech59965.2025.11180484.

[3] J. D. Rios-Peñaloza, A. Mortera-Canga, and M. Prodanovic, "Dynamic Stability Considerations During Capacity Expansion Planning," 2025 IEEE International Conference on Environment and Electrical Engineering and 2025 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe), Chania, Crete, Greece, July 2025, DOI:10.1109/EEEIC/ICPSEurope64998.2025.11169185.

[4] J. Hörsch, F. Hofmann, D. Schlachtberger, and T. Brown, "PyPSA-Eur: An open optimisation model of the European transmission system," Energy Strategy Reviews, vol. 22, pp. 207-215, 2018, ISSN 2211-467X, DOI:10.1016/j.esr.2018.08.012.

[5] S. Krahmer, S. J. Rasti, L. Fiedler, and M. Schmidt, "PowerFactory-Tools: A Python Package to Facilitate the Control of DIgSILENT PowerFactory," Journal of Open Source Software, vol. 10(116), 9281, 2025, DOI:10.21105/joss.09281.

[6] powfacpy. [Online]. Available: https://pypi.org/project/powfacpy/.

[7] R. Bolgaryn et al., "Recent Developments in Open Source Simulation Software pandapower and pandapipes," 2022 Open Source Modelling and Simulation of Energy Systems (OSMSES), Aachen, Germany, 2022, DOI:10.1109/OSMSES54027.2022.9769084.

[8] I. Cußmann et al., "Ein offenes netzebenen-und sektorenübergreifendes Planungsinstrument zur Bestimmung des optimalen Einsatzes und Ausbaus von Flexibilitätsoptionen in Deutschland: Projektabschlussbericht: eGon," Zentrum für nachhaltige Energiesysteme (ZNES), Hochschule Flensburg, 2024.

[9] Übertragungsnetzbetreiber, "Netzentwicklungsplan Strom 2037/2045 (2025), erster Entwurf," 2025.

[10] "Langfristszenarien – Wissenschaftliche Analysen zur Dekarbonisierung Deutschlands." [Online]. Available: https://langfristszenarien.de/.

[11] J. Nitsch et al., "Langfristszenarien und Strategien für den Ausbau der erneuerbaren Energien in Deutschland bei Berücksichtigung der Entwicklung in Europa und global – Schlussbericht," 2012. [Online]. Available: https://www.researchgate.net/profile/Michael-Sterner/publication/259895385_Langfristszenarien_und_Strategien_fur_den_Ausbau_der_erneuerbaren_Energien_in_Deutschland_bei_Berucksichtigung_der_Entwicklung_in_Europa_und_global/links/570c089e08aee0660351aa71/Langfristszenarien-und-Strategien-fuer-den-Ausbau-der-erneuerbaren-Energien-in-Deutschland-bei-Beruecksichtigung-der-Entwicklung-in-Europa-und-global.pdf.

[12] Vereinigung der Fernleitungsnetzbetreiber Gas e.V¿ (FNBGas), "Planungsstand Wasserstoff-Kernnetz vom 12.07.2023,", 2023. [Online]. Available: https://fnb-gas.de/wp-content/uploads/2023/07/2023-07-12_FNB-Gas_Planungsstand-Wasserstoff-Kernnetz.pdf.

[13] EHB initiative, "European Hydrogen Backbone: Boosting EU Resilience and Competitiveness," November 20, 2024. [Online]. Available: https://www.ehb.eu/files/downloads/1732103116_EHB-Boosting-EU-Resilience-and-Competitiveness-20-11-VF.pdf.

[14] Gurobi Optimization LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com.

[15] ENTSO-E System Operations Committee, "LFC Annual Report 2024," Sep. 2025. [Online]. Available: https://eepublicdownloads.entsoe.eu/clean-documents/SOC%20documents/LFC/20260211_ALFC_report_2024_updated.pdf.

[16] VDE, "Erzeugungsanlagen am Niederspannungsnetz – Technische Mindestanforderungen für Anschluss und Parallelbetrieb von Erzeugungsanlagen am Niederspannungsnetz (TAR Niederspnung)," VDE-AR-N 4105:2018-11, October 2020, Berlin.

[17] VDE, "Technische Regeln für den Anschluss von Kundenanlagen an das Mittelspannungsnetz, deren Betrieb (TAR Mittelspannung)," VDE-AR-N 4110:2023-09, September 2023, Berlin.

[18] 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, DOI:10.3390/su12239899.

[19] R. Bolgaryn et al., "Recent developments in open source simulation software pandapower and pandapipes," 2022 Open Source Modelling and Simulation of Energy Systems (OSMSES), 2022, DOI:10.1109/OSMSES54027.2022.9769084.

[20] R. Bolgaryn et al., "Open source simulation software pandapower and pandapipes: recent developments," 2023 Open Source Modelling and Simulation of Energy Systems (OSMSES), 2023, DOI:10.1109/OSMSES58477.2023.10089685.