Python for Power System Analysis (PyPSA): Free Software for Planning Energy Systems with High Shares of Renewables

Tom Brown, Jonas Hörsch, David Schlachtberger
Frankfurt Institute for Advanced Studies, Ruth-Moufang-Straße 1, 60438 Frankfurt am Main, Germany
Email: brown@fias.uni-frankfurt.de

***Abstract*—Python for Power System Analysis (PyPSA) is a free software toolbox for simulating and optimising modern power systems over multiple periods. PyPSA includes models for conventional generators with unit commitment, variable wind and solar generation, storage units, coupling to other energy sectors, and mixed alternating and direct current networks. It is designed to be easily extensible and to scale well with large networks and long time series. In this paper the basic functionality of PyPSA is described, including the formulation of the full power flow equations and the multi-period optimisation of operation and investment with linear power flow equations. PyPSA is positioned in the existing free software landscape as a bridge between traditional power flow analysis tools for steady-state analysis and full multi-period energy system models. The functionality is demonstrated on two open datasets of the transmission system in Germany (based on SciGRID) and Europe (based on GridKit).**

I. Introduction

Power system tools model the interactions between the electrical grid and the consumers and generators which use the grid. The importance of software modelling of the grid has risen in recent years given the increase in distributed and fluctuating wind and solar generation, and the increasing electrification of all energy demand. On the generation side, variable renewable generation causes loading in parts of the grid where it was never expected, and introduces new stochastic influences on the flow patterns. On the demand side, the need to decarbonise the transport and heating sectors is leading to the electrification of these sectors and hence higher electrical demand, replacing internal combustion engines with electric motors in the transport sector, and replacing fossil fuel boilers with heat pumps, resistive heaters and cogeneration for low-temperature space and water heating. In addition, the increasing deployment of storage technologies introduces many users which are both consumers and generators of energy.

The increasing complexity of the electricity system requires new tools for power system modelling. Many of the tools currently used for power system modelling were written in the era before widespread integration of renewable energy and the electrification of transport and heating. They therefore typically focus on network flows in single time periods. Examples of such tools include commercial products like DIgSILENT PowerFactory [1], PSS/E [2], PSS/SINCAL [3], NEPLAN [4] and PowerWorld [5], and open tools such as PSAT [6], MATPOWER [7], PYPOWER [8] and pandapower [9] (see [10] for a full list of power system analysis tools).

The consideration of multiple time periods is important on the operational side for unit commitment of conventional generators and the optimisation of storage and demand side management, and on the investment side for optimising infrastructure capacities over representative load and weather situations. Several tools have these capabilities, such as PLEXOS [11], TIMES [12], PRIMES [13], OSeMOSYS [14], oemof [15], minpower [16], calliope [17], and urbs [18], but their representations of electrical grids are often simplified.

Python for Power System Analysis (PyPSA) [19], the tool presented in this paper, was developed at the Frankfurt Institute for Advanced Studies to bridge the gap between power system analysis software and general energy system modelling tools. PyPSA can model the operation and optimal investment of the energy system over multiple periods. It has models for the unit commitment of conventional generators, time-varying wind and solar generators, storage units, all combinations of direct and alternating current electricity networks, and the coupling of electricity to other energy sectors, such as gas, heating and transport. It can perform full load flow calculations and linearised optimal load flow, including under consideration of security constraints. It was written from the start with variable renewables, storage and sector-coupling in mind, so that it performs well with large networks and long time series.

Given the complexity of power system tools and the different needs of different users, it is crucial that such tools are both transparent in what they do and easily extendable by the user. To this end, PyPSA was released as free software, which means that the user is free to inspect and modify the code, provided that if they publish modified software, they also release the code changes back to the community. Free software and open data also guarantee that research results can be reproduced by any third party, which is important given the large investment decisions that will need to be made on the basis of energy system modelling to reduce greenhouse gas emissions and combat global warming [20]. PyPSA is already used by more than a dozen research institutes worldwide, and the website has been visited by people from over 120 countries. As of June 2017 it has been used in five research papers [21], [22], [23], [24], [25].

This paper describes version 0.9.0 of PyPSA [19]. In Section II the mathematical functionality of PyPSA is described. PyPSA is compared with other software in Section III. Several example applications are given in IV before conclusions are drawn in V.

TABLE I
PYPSA COMPONENTS

Network	Container for all other network components.
Bus	Fundamental nodes to which all other components attach.
Load	A consumer of energy.
Generator	Generator whose feed-in can be flexible subject to minimum loading or minimum down and up times, or variable according to a given time series of power availability.
Storage Unit	A device which can shift energy from one time to another, subject to efficiency losses.
Store	A more fundamental storage object with no restrictions on charging or discharging power.
Shunt Impedance	An impedance in shunt to a bus.
Line	A branch which connects two buses of the same voltage.
Transformer	A branch which connects two buses of different voltages.
Link	A branch with a controllable power flow between two buses.

II. FUNCTIONALITY

In this section the basic components, power flow, linear optimal power flow, energy system optimisation and other functionality of PyPSA are described.

A. Components

PyPSA's representation of the power system is built by connecting the components listed in Table I.

Buses are the fundamental nodes to which all other components attach. Their mathematical role is to enforce energy conservation at the bus at all times (essentially Kirchhoff's Current Law).

Loads, generators, storage units, stores and shunt impedances attach to a single bus and determine the power balance at the bus. Loads represent a fixed power demand; a generator's dispatch can be optimised within its power availaiblity; stores can shift power from one time to another with a standing loss efficiency for energy leakage; storage units behave like stores, but they can also have efficiency losses and power limits upon charging and discharging; finally shunt impedances have a voltage-dependent power consumption.

Lines and transformers connect two buses with a given impedance. Power flows through lines and transformers according to the power imbalances at the buses and the impedances in the network. Lines and transformers are referred to collectively as 'passive branches' to distinguish them from controllable link branches. The impedances of the passive branches are modelled internally using the equivalent PI model. The relation between the series impedance $z = r + jx$, the shunt admittance $y = g + jb$, the transformer tap ratio τ, the transformer phase shift θ^{shift}, and the currents I_0, I_1 and voltages V_0, V_1 at the buses labelled 0 and 1 is given by

$$\begin{pmatrix} I_0 \\ I_1 \end{pmatrix} = \begin{pmatrix} \frac{1}{z} + \frac{y}{2} & -\frac{1}{z}\frac{1}{\tau e^{-j\theta^{\text{shift}}}} \\ -\frac{1}{z}\frac{1}{\tau e^{j\theta^{\text{shift}}}} & \left(\frac{1}{z} + \frac{y}{2}\right)\frac{1}{\tau^2} \end{pmatrix} \begin{pmatrix} V_0 \\ V_1 \end{pmatrix} \quad (1)$$

The equivalent circuit is shown in Figure 1. This circuit is for the case where the tap-changer is on the primary side; a similar equation and figure for the case where the tap-changer is on the secondary side is given in the documentation [19]. The line model defaults to the PI model, while the transformer model defaults to the more accurate T model, which is converted to the PI model using the standard delta-wye transformation. For convenience standard types for lines and transformers in networks at 50 Hz are provided following the conversion formula from nameplate parameters to impedances and the typical parameters provided in pandapower [9], so that the user does not have to input the impedances manually. The typical parameters in pandapower are based on [26], [27], [28].

Links connect two buses with a controllable active power dispatch that can be set by the user or optimised by PyPSA. Links can be used to represent point-to-point high voltage direct current (HVDC) lines, import-export capacities in transport models such as Net-Transfer-Capacity (NTC) models, or general energy conversion processes with a given efficiency, such as resistive heaters or heat pumps (from electricity to heat) or gas boilers (from gas to heat). Their efficiency can also be time-varying (e.g. to represent the ambient temperature dependence of a heat pump's coefficient of performance). [Networks of links implement Kirchoff's Current Law (energy conservation at each bus), but not Kirchoff's Voltage Law, which is obeyed by networks of passive branches.]

B. Power flow without optimisation

In a power flow calculation, the user specifies the power dispatch of all dispatchable components (loads, generators, storage units, stores and links) and then PyPSA computes the resulting voltages in the network and hence the power flows in passive branches (lines and transformers) based on their impedances.

1) Power flow equations for AC networks: A power flow calculation for an alternating current (AC) network ensures that for all buses labelled by n we have

$$S_n = V_n I_n^* = \sum_m V_n Y_{nm}^* V_m^* \quad (2)$$

where $S_n = P_n + jQ_n$ is the apparent power injected at the bus, I_n is the complex current and $V_n = |V_n|e^{j\theta_n}$ is the complex voltage, whose rotating angle is measured relative to a chosen slack bus. Y_{nm} is the bus admittance matrix, which is constructed for all buses based on the contributions from the individual branch admittance matrices from equation (1).

The inputs and outputs for the buses are given as follows:

- For the chosen slack bus $n = 0$, it is assumed that the voltage magnitude $|V_0|$ and the voltage angle θ_0 are given. PyPSA must find the powers P_0 and Q_0.
- For PQ buses, P_n and Q_n are given; $|V_n|$ and θ_n are to be found.
- For PV buses, P_n and $|V_n|$ are given; Q_n and θ_n are to be found.

The non-linear equation system (2) is then solved using the Newton-Raphson algorithm [29] and, by default, an initial 'flat' guess of $\theta_n = \theta_0$ and $|V_n| = 1$ (per unit). The initial guess can also be specified ('seeded') by the user, using for example the linearised power flow solution.

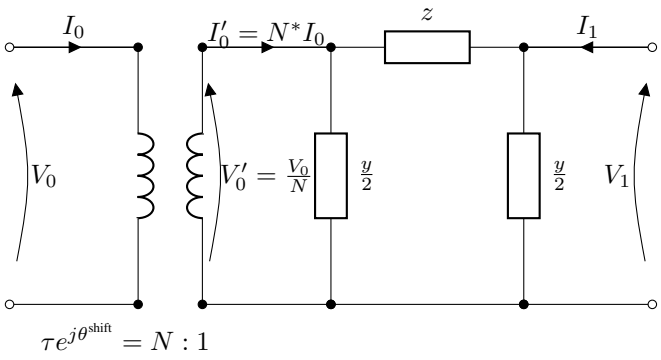

Fig. 1. Electrical property definitions for passive branches (lines and transformers).

2) Power flow equations for DC networks: A power flow calculation for a direct current (DC) network ensures that for all buses labelled by n we have

$$P_n = V_n I_n = \sum_m V_n G_{nm} V_m \tag{3}$$

where P_n is the active power injected at the bus and the voltage, current and the conductance matrix G_{ij} are now all real quantities. This non-linear equation is also solved with the Newton-Raphson algorithm.

3) Linearised power flow equations for AC networks: In some circumstances a linearisation of the AC power flow equations (2) can provide a good approximation to the full non-linear solution [30], [31]. The linearisation is restricted to calculating active power flows based on voltage angle differences and branch series reactances. It assumes that reactive power flow decouples from active power flow, that there are no voltage magnitude variations, voltage angles differences across branches are small enough that $\sin\theta \sim \theta$ and branch resistances are negligible compared to branch reactances. This makes it suitable for short overhead transmission lines close to their natural loading.

In this case it can be shown [7] that the voltage angles are related to the active power injections by a matrix

$$P_n = \sum_m (KBK^T)_{nm}\theta_m - \sum_\ell K_{n\ell} b_\ell \theta_\ell^{\text{shift}} \tag{4}$$

where K is the incidence matrix of the network, B is the diagonal matrix of inverse branch series reactances x_ℓ multiplied by the tap ratio τ_ℓ, i.e. $B_{\ell\ell} = b_\ell = \frac{1}{x_\ell \tau_\ell}$, and $\theta_\ell^{\text{shift}}$ is the phase shift for a transformer. The matrix KBK^T is singular with a single zero eigenvalue for a connected network and can be inverted by first deleting the row and column corresponding to the slack bus.

4) Linearised power flow equations for DC networks: For DC networks the equation (3) is linearised by positing $V_n = 1 + \delta V_n$ and assuming that δV_n is small. The resulting equations mirror the linearised AC approximation with the substitutions $\theta_n \to \delta V_n$ and $x_\ell \to r_\ell$.

C. Optimisation with linear power flow equations

PyPSA can optimise both operation and investment in the energy system as a linear problem using the linear power flow equations.

PyPSA minimises total system costs, which include the variable and fixed costs of generation, storage and transmission, given technical and physical constraints. The objective function is given by

$$\min_{\substack{G_{n,s},F_\ell,\\ g_{n,s,t},f_{\ell,t}}} \left[\sum_{n,s} c_{n,s} G_{n,s} + \sum_\ell c_\ell F_\ell + \sum_{n,s,t} w_t o_{n,s} g_{n,s,t}\right] \tag{5}$$

and consists of the capacities $G_{n,s}$ at each bus n for generation and storage technologies s and their associated annuitised fixed costs $c_{n,s}$, the dispatch $g_{n,s,t}$ of the unit at time t and the associated variable costs $o_{n,s}$, and the branch capacities F_ℓ for each branch ℓ and their annuitised fixed costs c_ℓ. The optimisation is run over multiple time periods t representing different weather and demand conditions. Each period can have a weighting w_t; the investment costs must then be annuitised for the total period $\sum_t w_t$ (typically a full year).

The dispatch of conventional generators $g_{n,s,t}$ is constrained by their capacity $G_{n,s}$

$$0 \le g_{n,s,t} \le G_{n,s} \qquad \forall\, n, s, t \tag{6}$$

Unit commitment for conventional generators is described in the documentation [19] and a forthcoming paper [32].

The maximum producible power of renewable generators depends on the weather conditions, which is expressed as an availability $\bar{g}_{n,s,t}$ per unit of its capacity:

$$\tilde{g}_{n,s,t} G_{n,s} \le g_{n,s,t} \le \bar{g}_{n,s,t} G_{n,s} \qquad \forall\, n, s, t \tag{7}$$

Curtailment may also be limited by introducing a lower bound on dispatch $\tilde{g}_{n,s,t}$.

The energy levels $e_{n,s,t}$ of all storage units have to be consistent between all hours and are limited by the storage energy capacity $E_{n,s}$

$$\begin{aligned} e_{n,s,t} = {} & \eta_{n,s,0}^{w_t} e_{n,s,t-1} \\ & + \eta_{n,s,+} w_t \left[g_{n,s,t}\right]^+ - \eta_{n,s,-}^{-1} w_t \left[g_{n,s,t}\right]^- \\ & + w_t g_{n,s,t,\text{inflow}} - w_t g_{n,s,t,\text{spillage}} \\ \tilde{e}_{n,s,t} E_{n,s} \le {} & e_{n,s,t} \le \bar{e}_{n,s,t} E_{n,s} \qquad \forall\, n, s, t \end{aligned} \tag{8}$$

Positive and negative parts of a value are denoted as $[\cdot]^+ = \max(\cdot, 0)$, $[\cdot]^- = -\min(\cdot, 0)$. The storage units can have a standing loss (leakage rate) $\eta_{n,s,0}$, a charging efficiency $\eta_{n,s,+}$, a discharging efficiency $\eta_{n,s,-}$, inflow (e.g. river inflow in a reservoir) and spillage. The initial energy level can be set by the user, or it is assumed to be cyclic, i.e. $e_{n,s,t=0} = e_{n,s,t=T}$. The store component is a more basic version of the storage unit: it's charging and discharging power cannot be limited and there are no charging and discharging efficiencies $\eta_{n,s,+}, \eta_{n,s,-}$. The energy levels of the store can also be restricted by time series $\tilde{e}_{n,s,t}, \bar{e}_{n,s,t}$ given per unit of the energy capacity $E_{n,s}$; this allows the demand-side management model of [33] to be implemented in PyPSA.

CO_2 emissions can be limited by a cap CAP_{CO2}, implemented using the specific emissions e_s in CO_2-tonne-per-MWh of the fuel s and the efficiency $\eta_{n,s}$ of the generator:

$$\sum_{n,s,t} \frac{1}{\eta_{n,s}} w_t g_{n,s,t} \cdot e_s \le \text{CAP}_{CO2} \quad \leftrightarrow \quad \mu_{CO2} \tag{9}$$

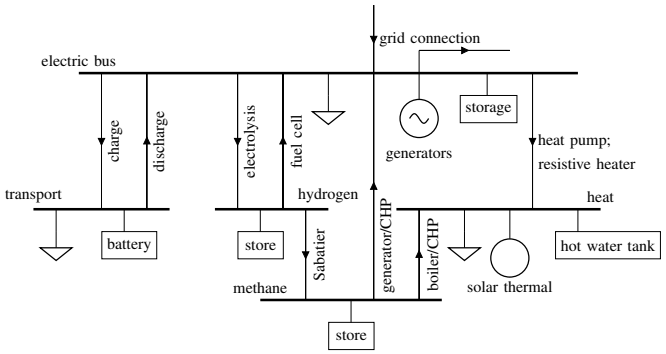

Fig. 2. Example of the coupling in PyPSA between electricity (at top) and other energy sectors: transport, hydrogen, natural gas and heating. There is a bus for each energy carrier, to which different loads, sources and converters are attached.

μ_{CO2} is the shadow price of this constraint.

The (inelastic) electricity demand $d_{n,t}$ at each bus n must be met at each time t by either local generators and storage or by the flow $f_{\ell,t}$ from a branch ℓ

$$\sum_s g_{n,s,t} - d_{n,t} = \sum_\ell K_{n\ell}\, \eta_{\ell,n,t}\, f_{\ell,t} \quad \leftrightarrow \quad \lambda_{n,t} \quad \forall\, n,t \tag{10}$$

where $K_{n\ell}$ is the incidence matrix of the network, $\eta_{\ell,n,t}$ is a factor for the efficiency of the energy transfer in ℓ and $\lambda_{n,t}$ is the marginal price at the bus. This equation implements Kirchhoff's Current Law (KCL).

The flows in all branches are constrained by their capacities F_ℓ

$$|f_{\ell,t}| \leq F_\ell \qquad \forall\, \ell,t \tag{11}$$

The flows in links are controllable. For links $\eta_{\ell,n,t} = 1$ if the link starts at bus n and takes a user-set time-dependent efficiency $\eta_{\ell,n,t} = \eta_{\ell,t}$ if the link ends at bus n.

Power flows in networks of passive branches (lines and transformers) according to the linear power flow equation (4). The security-constrained LOPF using Line Outage Distribution Factors (LODFs) is described in the documentation [19] and a forthcoming paper [32].

D. Coupling to other energy sectors

PyPSA can also optimise operation and investment in other energy sectors, such as natural gas, heating and transport. These sectors can be modelled using a network of links with efficiencies for energy conversion losses; an example from a recent paper [22] is shown in Figure 2. For example, links from electricity to heat buses can represent resistive heaters and/or heat pumps (the latter can also be modelled with a time-dependent coefficient of performance, given the importance of capturing the dependence of heat pump performance on outside temperature [37]). Combined Heat and Power plants (CHPs) can also be modelled by adding additional constraints for the back pressure and fuel consumption (see the PyPSA examples [19]). Depletable resources such as natural gas are modelled with stores.

E. Network clustering

PyPSA also implements a variety of network clustering algorithms to reduce the number of buses in a network while preserving important transmission lines. For example, the k-means clustering algorithm was recently used in [25] to examine the effect of clustering on investment optimisation results.

III. Comparison to other free software power system tools in Python

Given the proliferation of free software tools available for modelling power systems, here a guide is provided that briefly compares PyPSA to other Python-based power system tools.

PYPOWER [8] is a port of an older version of MATPOWER [7] from Matlab to Python. It does not make strong use of Python's object-oriented interface and structures data using NumPy arrays, which makes it difficult to track component attributes. It has no functionality to deal with multi-period OPF, which makes it unsuitable for unit commitment, storage optimisation or investment optimisation. This reflect the functionality of older versions of MATPOWER, but the latest version 6.0 of MATPOWER includes the MATPOWER Optimal Scheduling Tool (MOST) [51], which does multi-period OPF. Unlike PyPSA, PYPOWER has the ability to do full non-linear OPF for single snapshots. Both PyPSA and PYPOWER are computationally faster than MATPOWER (in some cases by a factor of 3) for solving load flow problems, thanks to the speed of the sparse linear algebra solver from SciPy [44], which interfaces to the C library umfpack [50].

pandapower [9] provides a pandas [42] interface to PYPOWER [8], which makes it easier to use, and adds useful functionality such as standard types (on which PyPSA's standard types are based), short circuit calculations, state estimation, and modelling of switches and three-winding transformers. The last four functions are currently missing in PyPSA, along with non-linear OPF, but like PYPOWER, pandapower does not have multi-period OPF functionality. pandapower is under active development and the PyPSA team stays in contact with the pandpower team to exchange tips and features, which is a clear benefit for both developers and users of free software.

PyPSA differs from more general energy system models such as OSeMOSYS [14], oemof [15], calliope [17] and urbs [18] by offering more detailed modelling of power networks, in particular the physics of power flow according to the impedances in the network. PyPSA can imitate a more general energy network using link components (see Section II-D), but cannot, for example, yet do the multi-year dynamic investment that OSeMOSYS does.

IV. Demonstration of features on the SciGRID and GridKit datasets

On the PyPSA website [19] a large number of examples of code using PyPSA has been uploaded for reference and to help users just starting out with the software. These range from basic small-scale networks demonstrating the features of PyPSA, to full transmission network models available as open data from the SciGRID [52] and GridKit projects [53], [54] which we demonstrate here.

The SciGRID model of Germany provides geo-referenced data for substations and transmission lines (220 kV and above). In one code example, data from openly-available sources on power plant locations and capacities, load distribution and time series are added to the SciGRID data so

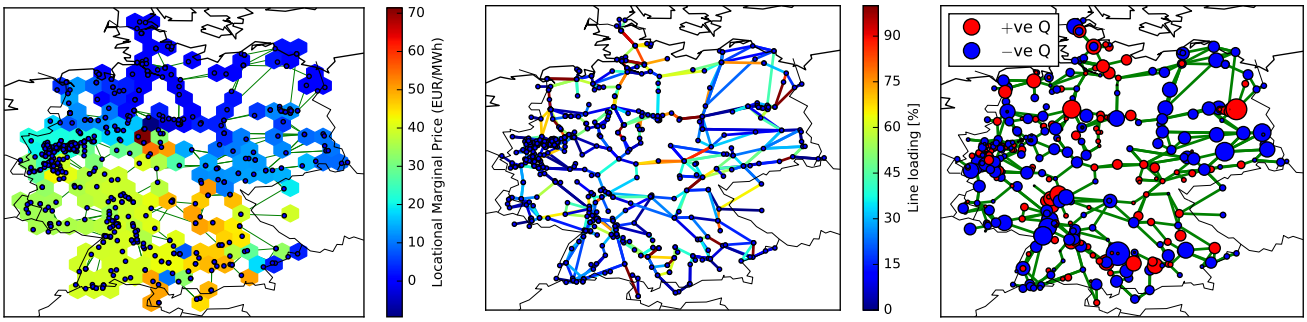

Fig. 3. *Left:* Locational marginal prices ($\lambda_{n,t}$ from equation (10)) for Germany in an hour with high wind and low load; *Middle:* Line loading during this hour: highly loaded lines in the middle of Germany prevent the transport of cheap wind energy to consumers in the South; *Right:* Reactive power feed-in necessary to keep all buses at unit nominal voltage.

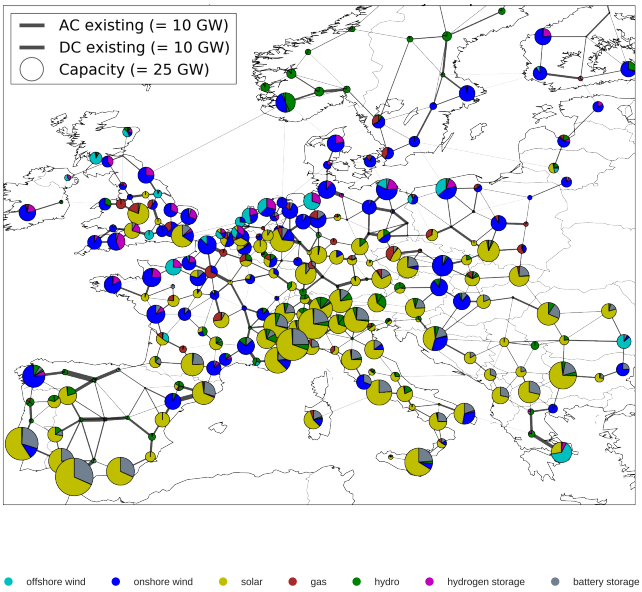

Fig. 4. Example of results of optimisation of generation and storage technologies in Europe to reduce CO_2 emissions in the European electricity sector by 95% compared to 1990 levels [25]. The grid topology is based on the GridKit network for Europe, clusteredfrom 5000 buses to 256 buses.

that load flow calculations can be carried out. The results of one such simulation for Germany with nodal pricing is shown in Figure 3. In this snapshot there was a large amount of zero-marginal-cost wind feed-in suppressing the locational marginal prices ($\lambda_{n,t}$ from equation (10)) in the North of Germany. Transmission bottlenecks in the middle of Germany prevent the transportation of this cheap electricity to the South, where more expensive conventional generators set the price. The linearly-optimised dispatch was then fed into a full non-linear power flow calculation where each bus was set to maintain nominal voltage; the resulting reactive power feed-in is also shown in Figure 3.

The data quality for the transmission grid in OpenStreetMap outside Germany is not of uniform quality, so for the European grid, an extract of the ENTSO-E interactive map [55] was made [53] using GridKit [54]. The details of how load, conventional power plants and renewable generation time series and expansion potentials were added to the grid data are detailed in a forthcoming paper [56]. The result of generation and storage investment optimisation for a clustering of the network from 5000 buses down to 256 buses, allowing no grid expansion and assuming a CO2 reduction of 95% compared to 1990 levels, is shown in Figure 4. The lack of grid expansion forces some balancing of renewable variability locally with storage. Short-term battery storage (grey) combines with solar power (yellow) in Southern Europe, while longer-term hydrogen storage (purple) pairs with wind power (blue) in Northern Europe. This system has an average cost of € 82/MWh. If the grid is optimally expanded, much of the storage can be eliminated and costs are as low as € 65/MWh [25].

V. Conclusions

In this paper a new toolbox has been presented for simulating and optimising power systems. Python for Power System Analysis (PyPSA) provides components to model variable renewable generation, conventional power plants, storage units, coupling to other energy sectors and multiply-connected AC and DC networks over multiple periods for the optimisation of both operation and investment. Tools are also provided for steady-state analysis with the full load flow equations. PyPSA's performance for large datasets, comparisons with other software packages and several example applications are demonstrated.

As free software, the code of PyPSA can easily be inspected and extended by users, thereby contributing to further research and also transparency in power system modelling. Given that public acceptance of new infrastructure is often low, it is hoped that transparent modelling can contribute to public understanding of the various options we face when designing a sustainable energy system.

Acknowledgments

This research was conducted as part of the CoNDyNet project, which is supported by the German Federal Ministry of Education and Research under grant no. 03SF0472C. The responsibility for the contents lies solely with the authors.

References

[1] DIgSILENT GmbH, "PowerFactory," http://digsilent.de/, 2017.
[2] Siemens AG, "PSS/E," http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/software-solutions/planning-data-management-software/planning-simulation/Pages/PSS-E.aspx, 2017.

[3] ——, "PSS/SINCAL," http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/software-solutions/planning-data-management-software/planning-simulation/pss-sincal/pages/pss-sincal.aspx, 2017.

[4] NEPLAN AG, "PowerFactory," http://www.neplan.ch/, 2017.

[5] PowerWorld Corporation, "PowerWorld," https://www.powerworld.com/, 2017.

[6] Federico Milano, "Power System Analysis Tool (PSAT)," http://faraday1.ucd.ie/psat.html, 2017.

[7] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "MATPOWER: Steady-state operations, planning and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, p. 12, 2011.

[8] Richard Lincoln, "PYPOWER," https://github.com/rwl/PYPOWER, 2017.

[9] L. Thurner, A. Scheidler, J. Dollichon, F. Schäfer, J.-H. Menke, F. Meier, S. Meinecke *et al.*, "pandapower - convenient power system modelling and analysis based on pypower and pandas," University of Kassel and Fraunhofer Institute for Wind Energy and Energy System Technology, Tech. Rep., 2016. [Online]. Available: http://pandapower.readthedocs.io/en/v1.2.2/_downloads/pandapower.pdf

[10] Open Electrical Wiki, "Power Systems Analysis Software," http://www.openelectrical.org/wiki/index.php?title=Power_Systems_Analysis_Software, 2017.

[11] Energy Exemplar, "PLEXOS," http://energyexemplar.com/, 2017.

[12] R. Loulou, U. Remne, A. Kanudia, A. Lehtila, and G. Goldstein, "Documentation for the TIMES Model - PART I 1–78," ETSAP, Tech. Rep., 2005.

[13] "The PRIMES Model," http://www.e3mlab.ntua.gr/, NTUA, Tech. Rep., 2009.

[14] M. Howells, H. Rogner, N. Strachan, C. Heaps, H. Huntington, S. Kypreos, A. Hughes, S. Silveira, J. DeCarolis, M. Bazillian, and A. Roehrl, "Osemosys: The open source energy modeling system: An introduction to its ethos, structure and development," *Energy Policy*, vol. 39, no. 10, pp. 5850 – 5870, 2011, sustainability of biofuels. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0301421511004897

[15] S. Hilpert, S. Günther, C. Kaldemeyer, U. Krien, G. Plessmann, F. Wiese, and C. Wingenbach, "Addressing energy system modelling challenges: The contribution of the open energy modelling framework (oemof)," *Preprints*, 2017.

[16] A. Greenhall and R. Christie, "Minpower: A power systems optimization toolkit," in *2012 IEEE Power and Energy Society General Meeting*, July 2012, pp. 1–6.

[17] S. Pfenninger, "Dealing with multiple decades of hourly wind and {PV} time series in energy models: A comparison of methods to reduce time resolution and the planning implications of inter-annual variability," *Applied Energy*, vol. 197, pp. 1 – 13, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261917302775

[18] J. Dorfner, M. Dorfner, S. Candas, S. Müller, Y. Özsahin, T. Zipperle, S. Herzog, Icedkk, and WYAUDI, "urbs: v0.6," Aug. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.60484

[19] T. Brown, J. Hörsch, and D. Schlachtberger, *Python for Power System Analysis (PyPSA) Version 0.9.0*, Apr. 2017, https://doi.org/10.5281/zenodo.582307. [Online]. Available: https://github.com/FRESNA/PyPSA

[20] S. Pfenninger, "Energy scientists must show their workings," *Nature*, vol. 542, p. 393, 2017.

[21] J. G. Dedecca, R. A. Hakvoort, and P. M. Herder, "Transmission expansion simulation for the European Northern Seas offshore grid," *Energy*, vol. 125, pp. 805 – 824, 2017.

[22] T. Brown, D. Schlachtberger, A. Kies, and M. Greiner, "Sector coupling in a simplified model of a highly renewable European energy system," in *Proceedings of 15th Wind Integration Workshop*, 2016.

[23] J. Hörsch, H. Ronellenfitsch, D. Witthaut, and T. Brown, "Linear Optimal Power Flow Using Cycle Flows," *ArXiv e-prints*, Apr. 2017. [Online]. Available: https://arxiv.org/abs/1704.01881

[24] D. Schlachtberger, T. Brown, S. Schramm, and M. Greiner, "The benefits of cooperation in a highly renewable European electricity network," *Energy*, vol. 134, pp. 469 – 481, 2017.

[25] Hörsch, J. and Brown, T., "The role of spatial scale in joint optimisations of generation and transmission for European highly renewable scenarios," in *Proceedings of 14th International Conference on the European Energy Market (EEM 2017)*, 2017. [Online]. Available: https://arxiv.org/abs/1705.07617

[26] K. Heuck, K.-D. Dettmann, and D. Schulz, *Elektrische Energieversorgung*, 9th ed. Berlin Heidelberg New York: Springer-Verlag, 2013.

[27] B. Oswald and D. Oeding, *Elektrische Kraftwerke und Netze.*, 6th ed. Berlin Heidelberg New York: Springer-Verlag, 2004.

[28] B. Oswald, "Vorlesung Elektrische Energieversorgung I: Skript Transformatoren," http://antriebstechnik.fh-stralsund.de/1024x768/Dokumentenframe/Versuchsanleitungen/EMA/Trafo.pdf, 2005.

[29] J. J. Grainger and W. D. Stevenson Jr., *Power System Analysis*. New York: McGraw-Hill, 1994.

[30] K. Purchala, L. Meeus, D. V. Dommelen, and R. Belmans, "Usefulness of DC power flow for active power flow analysis," in *IEEE Power Engineering Society General Meeting*, June 2005, pp. 454–459 Vol. 1.

[31] B. Stott, J. Jardim, and O. Alsac, "Dc power flow revisited," *IEEE Trans. Power Syst.*, vol. 24, no. 3, p. 1290, 2009.

[32] T. Brown, J. Hörsch, and D. Schlachtberger, "PyPSA: Python for Power System Analysis," forthcoming.

[33] D. Kleinhans, "Towards a systematic characterization of the potential of demand side management," *ArXiv e-prints*, Jan. 2014.

[34] Hagspiel, S., Jägemann, C., Lindenburger, D., Brown, T., Cherevatskiy, S., and Tröster, E., "Cost-optimal power system extension under flow-based market coupling," *Energy*, vol. 66, pp. 654–666, 2014.

[35] J. G. Dedecca, "Mixed integer modification of the pypsa package," 2017, https://github.com/jdedecca/MILP_PyPSA.

[36] L. Bahiense, G. C. Oliveira, M. Pereira, and S. Granville, "A mixed integer disjunctive model for transmission network expansion," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 560–565, Aug 2001.

[37] S. N. Petrović and K. B. Karlsson, "Residential heat pumps in the future Danish energy system," *Energy*, vol. 114, pp. 787 – 797, 2016.

[38] Wollenberg, B. and Wood, A., *Power Generation, Operation, and Control*. John Wiley & Sons, 1996.

[39] A. Trias, "The holomorphic embedding load flow method," in *Power and Energy Society General Meeting, 2012 IEEE*, July 2012, pp. 1–8.

[40] S. P. Vera, "Gridcal," 2017, https://github.com/SanPen/GridCal.

[41] "Python programming language, version 3.5," https://www.python.org/, 2017.

[42] W. McKinney, "Data structures for statistical computing in Python," in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 51–56. [Online]. Available: http://conference.scipy.org/proceedings/scipy2010/mckinney.html

[43] S. van der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, pp. 22–30, 2011.

[44] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online]. Available: http://www.scipy.org/

[45] W. E. Hart, J.-P. Watson, and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

[46] W. E. Hart, C. Laird, J.-P. Watson, and D. L. Woodruff, *Pyomo–optimization modeling in python*. Springer Science & Business Media, 2012, vol. 67.

[47] F. Pérez and B. E. Granger, "IPython: A System for Interactive Scientific Computing," *Computing in Science & Engineering*, vol. 9, pp. 21–29, 2007.

[48] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, pp. 90–95, 2007.

[49] C. Josz, S. Fliscounakis, J. Maeght, , and P. Panciatici, "Data in MATPOWER and QCQP Format: iTesla, RTE Snapshots, and PEGASE," *ArXiv e-prints*, Mar. 2016. [Online]. Available: https://arxiv.org/abs/1603.01533

[50] T. A. Davis, "Algorithm 832: Umfpack v4.3—an unsymmetric-pattern multifrontal method," *ACM Trans. Math. Softw.*, vol. 30, no. 2, pp. 196–199, Jun. 2004. [Online]. Available: http://doi.acm.org/10.1145/992200.992206

[51] C. E. Murillo-Sanchez, R. D. Zimmerman, C. L. Anderson, and R. J. Thomas, "Secure planning and operations of systems with stochastic sources, energy storage and active demand," *IEEE Transactions on Smart Grid*, vol. 4, pp. 2220–2229, 2013.

[52] C. Matke, W. Medjroubi, and D. Kleinhans, "SciGRID - An Open Source Reference Model for the European Transmission Network (v0.2)," Jul. 2016. [Online]. Available: http://www.scigrid.de

[53] B. Wiegmans, "GridKit extract of ENTSO-E interactive map," Jun. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.55853

[54] ——, "Improving the topology of an electric network model based on open data," Master's thesis, Energy and Sustainability Research Institute, University of Groningen, 2015.

[55] "ENTSO-E Interactive Transmission System Map," Jan. 2016. [Online]. Available: https://www.entsoe.eu/map/Pages/default.aspx

[56] J. Hörsch, F. Hofmann, D. Schlachtberger, and T. Brown, *PyPSA-EU: An open optimization model of the European transmission system*, 2017, in preparation.