ElectricGrid.jl - A Julia-based modeling and simulation tool for power electronics-driven electric energy grids

Oliver Wallscheid[1], **Sebastian Peitz**[2], **Jan Stenner**[2], **Daniel Weber**[1], **Septimus Boshoff**[1], **Marvin Meyer**[1], **Vikas Chidananda**[2], and **Oliver Schweins**[1]

1 Chair of Power Electronics and Electrical Drives, Paderborn University, Paderborn, Germany **2** Chair of Data Science for Engineering, Paderborn University, Paderborn, Germany

DOI: 10.21105/joss.05616

Software

- Review
- Repository
- Archive

Editor: Øystein Sørensen

Reviewers:

- @kiranshila
- @degleris1

Submitted: 28 April 2023
Published: 29 August 2023

Summary

The ElectricGrid.jl toolbox provides a transient simulation framework for electric energy grids based on power electronic converters. With a few lines of code, a parameterised electric grid model on component level can be initialised in Julia. An example grid is shown in Figure 1.

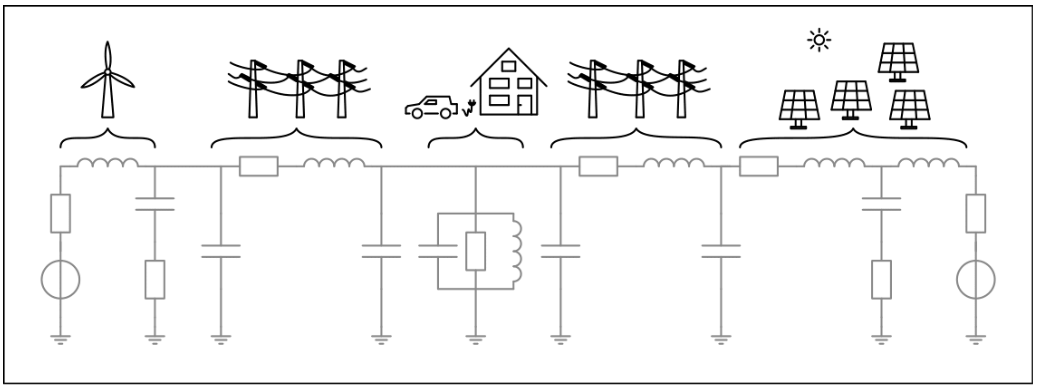

Figure 1: Exemplary electric energy grid in a simplified single phase representation.

By means of state-space models (set of first order ordinary differential equations), electrical energy grids can be constructed and simulated in a short time, which can be utilised for synthetic data generation (e.g., for data-driven meta-modelling) and to compare grid control methods.

Statement of need

Decentralised, electrical energy networks have special demands on operating and control procedures to ensure a continuous and efficient energy supply and simultaneously play an important role in the integration of renewable energy sources (Guerrero et al., 2013). This applies both in connection with conventional centralised power grids and for islanded microgrids in remote areas (Lund et al., 2017). Due to their high efficiency and flexibility, power electronic converters have become the standard tool for integrating renewable energy sources, energy storages and loads in electrical energy grids. The field of power electronics covers the application of solid-state electronics to the control and conversion of electric power, which is performed with semiconductor switching devices such as diodes or power transistors. This includes energy conversion in terms of voltage and current amplitude, frequency and phase angle, as well as the number of phases between two or more electrical energy systems to be connected.

Controlling (decentralised) electric grids is a challenging task due to their stochastic, heterogeneous and volatile characteristics (in particular regarding the connected loads). At the same time, high requirements are made with regard to aspects such as safety, quality and availability. This results in a high demand for comprehensive testing of new control concepts during their development phase and comparisons with the state of the art to ensure their feasibility. This applies in particular to emerging data-driven control approaches such as reinforcement learning (RL), the stability and operating behavior of which cannot be evaluated a priori (Garcıa & Fernández, 2015). Besides RL methods, being data-driven, result in a model-free and self-adaptive controller design with little human effort labeling them a promising tool for controlling unknown or changing systems targeting the above described challanges. However, there is a need for further research into the requirements for energy networks in terms of safety, robustness and availability before RL-based controllers can be used in real applications (Glavic et al., 2017; Zhang et al., 2018).

`ElectricGrid.jl` is a Julia package for setting up realistic electric grid simulations with support for control options. A number of parameters are made avaible to the user to evaluate the various control options. If no details are given, all parameters are generated automatically, either through randomness on a physically meaningful basis or by verified design methods. This enables both experts from the field of electrical energy networks to test certain configurations and experts from the field of artificial intelligence to test new control approaches without any prior knowledge of electrical engineering. Therefore, `ElectricGrid.jl` is designed to be used by students, academics, and industrial researchers in the field of simulation and data-driven analysis of electrical energy systems. The primary objective of the toolbox is to facilitate entry for new users into the modeling, control, and testing of small to large scale electric power grids and to provide a platform on which different control methods (including RL) can be compared under defined conditions (benchmarks).

The experiments are based on dynamic simulations in the time domain which allows for accurate control and test investigations during transients and steady state down to component level. This is an essential difference to already available open-source solutions for the simulation of electrical energy grids like PyPSA (Brown et al., 2017), Powermodels.jl (Coffrin et al., 2018) and pandapower (Thurner et al., 2018) which, in contrast, usually perform the calculations in a (quasi)-stationary state. Also these frameworks tend to focus on large-scale power systems at the transmission and distribution grid level, which does not allow an evaluation of, e.g., control on component level in case of load fluctuations in the grid. In addition, a few tools like the one presented in Lara et al. (2023) already exist in Julia, which offer dynamic simulations. However, the latter is based on different simplifications (e.g., assumption of a symmetric grid, fixed frequency, …) and also do not offer an interface to RL toolboxes. To ensure a seamless integration of the control algorithms, the Gymnasium-based API (Farama-Foundation, 2023) should be used, which has been established as a standard in recent years. For other projects implementing the Gymnasium-based API, such as GridAlive (based on Grid2Op (Donnot, 2020)) and ChroniX2Grid (Marot et al., 2020), the focus remains on top-down control and steady-state models. Therefore, `ElectricGrid.jl` provides a tool to close these highlighted gaps.

Interfaces for control and reinforcement learning

The API is designed to provide a user-friendly interface to connect a modeled electric energy grid with a wide range of classical control methods like shown in Figure 2.

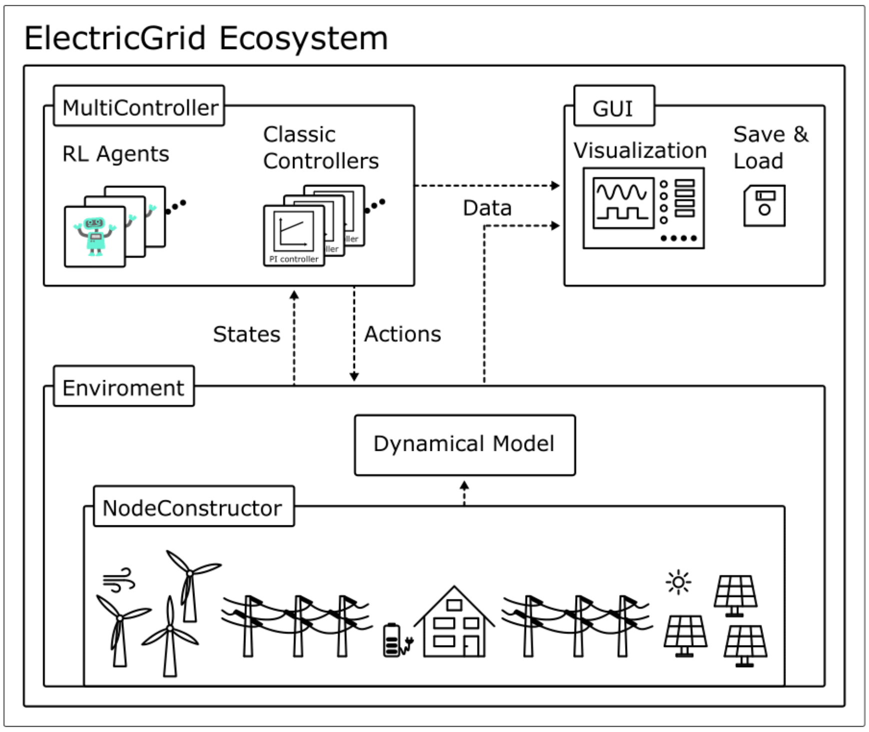

Figure 2: Overview of the functionality and interconnections of the ElectricGrid.jl framework.

Already provided are classic controllers (i.e., industry standard contollers) like linear feedback proportional integral (PI) in the direct-quadrature-zero (DQ0) rotating reference frame. These control methods can be used out of the box including automatical tuning procedures. Many basic auxiliary functionalities for the essential operation of electric power grids are provided too such as coordinate transformations for basic controller classes, data logging, measurement of real and imaginary powers, and phase-locked loops for frequency and phase angle extraction. The interface provided by Tian & contributors (2020) is also available for training data-driven control approaches like RL. This enables users who want to integrate contemporary open-source Julia-based RL toolboxes such as `ReinforcementLearning.jl` (Tian & contributors, 2020). Following this structure, nearly every control approach, including data-driven RL, can be implemented and tested with `ElectricGrid.jl` in a relatively short amount of time.

Features

The `ElectricGrid.jl` toolbox provides the following key features:

- Framework to set up an experiment with a parameterised energy grid in a few lines of code.
- Dynamic simulation of electricity grids on component level including single and multi-phase systems as well as AC and DC operation with arbitrary waveforms.
- Calculation, evaluation and logging of every single time step covering states, action and auxiliary quantities.
- Large variety of predefined and parameterisable controllers (droop, VSG, swing, active-reactive) are available.
- Interesting use cases applying data-driven learning.

Examples

For illustration and interactive introduction, Jupyter Notebooks are available for each topic. These provide clear and easy-to-expand examples of: - Utilising ElectricGrid.jl to build an energy grid, - Theoretical principles behind the calculations, - Applying classic controllers on the electrical grid, - Training an RL agent on the electrical grid.

Availability and installation

`ElectricGrid.jl` is supported and tested on Linux, Windows and macOS. The package should be installed using the Julia package manager. In a Julia terminal run the follwing:

```
import Pkg
Pkg.add("ElectricGrid")
```

Alternatively, it can also be installed from the Github source code. To do that, clone the repository, start Julia, activate the project by pressing `]` to access Pkg mode and then `activate path/to/ElectricGrid` or `activate .` If you started Julia in your ElectricGrid directory and afterwards run `instantiate`.

The source code, guide and examples are available on the GitHub repository (https://github.com/upb-lea/JuliaElectricGrid.jl).

Individual contributions of the authors

Following are shown the main fields of each individual contributor of ElectricGrid.jl:

- O. Wallscheid: Concept design and idea generation, testing and technical feedback, administrative project management
- S. Peitz: Administrative project management, concept-oriented feedback
- J. Stenner: API RL framework, API environment framework, basic system architecture
- D. Weber: Application examples, API environment framework, basic system architecture, unit tests
- S. Boshoff: Application examples, primary controllers in DQ0 frame, decentralised secondary controllers, Luenberger observers, stochastic processes, inverter filter design, cable design, unit tests
- M. Meyer: Basic system architecture, application examples, unit tests
- V. Chidananda: System analytics, unit tests
- O. Schweins: Basic system architecture

Acknowledgements

The authors would also like to acknowledge the funding and support of this work by the Federal Ministry of Education and Research of Germany (grant number 011S21064).

References

Brown, T., Hörsch, J., & Schlachtberger, D. (2017). PyPSA: Python for power system analysis. *arXiv Preprint arXiv:1707.09913*. https://doi.org/10.5334/jors.188

Coffrin, C., Bent, R., Sundar, K., Ng, Y., & Lubin, M. (2018). PowerModels.jl: An open-source framework for exploring power flow formulations. *2018 Power Systems Computation Conference (PSCC)*, 1–8. https://doi.org/10.23919/PSCC.2018.8442948

Donnot, B. (2020). Grid2op- A testbed platform to model sequential decision making in power systems. . In *GitHub repository*. https://GitHub.com/rte-france/grid2op; GitHub.

Farama-Foundation. (2023). Farama-foundation/gymnasium: A standard API for single-agent reinforcement learning environments, with popular reference environments and related utilities (formerly gym). In *GitHub*. https://github.com/Farama-Foundation/Gymnasium

Garcıa, J., & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, *16*(1), 1437–1480.

Glavic, M., Fonteneau, R., & Ernst, D. (2017). Reinforcement learning for electric power system decision and control: Past considerations and perspectives. *IFAC-PapersOnLine*, *50*. https://doi.org/10.1016/j.ifacol.2017.08.1217

Guerrero, J. M., Chandorkar, M., Lee, T.-L., & Loh, P. C. (2013). Advanced control architectures for intelligent microgrids—part i: Decentralized and hierarchical control. *IEEE Transactions on Industrial Electronics*, *60*(4), 1254–1262. https://doi.org/10.1109/TIE.2012.2194969

Lara, J. D., Henriquez-Auba, R., Ramasubramanian, D., Dhople, S., Callaway, D. S., & Sanders, S. (2023). Revisiting power systems time-domain simulation methods and models. *arXiv Preprint arXiv:2301.10043*. https://doi.org/10.1109/TPWRS.2023.3303291

Lund, H., Østergaard, P. A., Connolly, D., & Mathiesen, B. V. (2017). Smart Energy and Smart Energy Systems. *Energy*, *137*, 556–565. https://doi.org/10.1016/j.energy.2017.05.123

Marot, A., Megel, N., Renault, V., & Jothy, M. (2020). ChroniX2Grid - The Extensive PowerGrid Time-serie Generator. In *GitHub repository*. https://github.com/BDonnot/ChroniX2Grid; GitHub.

Thurner, L., Scheidler, A., Schäfer, F., Menke, J.-H., Dollichon, J., Meier, F., Meinecke, S., & Braun, M. (2018). Pandapower—an open-source python tool for convenient modeling, analysis, and optimization of electric power systems. *IEEE Transactions on Power Systems*, *33*(6), 6510–6521. https://doi.org/10.1109/TPWRS.2018.2829021

Tian, J., & contributors, other. (2020). *ReinforcementLearning.jl: A reinforcement learning package for the Julia programming language*. https://github.com/JuliaReinforcementLearning/ReinforcementLearning.jl

Zhang, D., Han, X., & Deng, C. (2018). Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems*, *4*(3), 362–370. https://doi.org/10.17775/CSEEJPES.2018.00520