Mourouvin, Rayane; Kukkola, Jarno; Tiitinen, Lauri; Hinkkanen, Marko

gritulator: Grid Converter Simulator in Python

Published in:
Proceedings of 2023 IEEE PES Innovative Smart Grid Technologies Europe, ISGT EUROPE 2023

DOI:
10.1109/ISGTEUROPE56780.2023.10407218

Published: 30/01/2024

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

gritulator: Grid Converter Simulator in Python

Rayane Mourouvin, Jarno Kukkola, Lauri Tiitinen, Marko Hinkkanen
Department of Electrical Engineering and Automation
Aalto University
Espoo, Finland
rayane.mourouvin@aalto.fi

***Abstract*—Open-source simulation tools are gaining interest in academia as they allow to share knowledge and experience between researchers. Furthermore, they are useful instruments for educational purposes. This paper presents gritulator, an open-source Python-based simulator, for grid converters. The simulator is available in GitHub at https://github.com/Aalto-Electric-Drives/gritulator. The structure of the simulator, and selected models and control methods are introduced. To illustrate the simulator in use, two converter-control examples are given and their simulated waveforms are compared with corresponding experimental measurements.**

***Index Terms*—Control systems, grid converters, open source, Python, time-domain simulation.**

I. Introduction

The current objectives to achieve a global energy transition towards renewable electrical systems up to 100% renewable energy-based grids is a tremendous challenge [1]. This global transition benefits from cooperation in order to spread the newly-developed solutions. Collaborative simulation tools are excellent assets to study hybrid grids where renewable energy sources, based on power electronics, and traditional power plants live together.

In the literature, there are open-source academic works, based on MATLAB, to model and simulate electric power systems, such as PSAT [2]. Other projects include FORTRAN-based RAMSES software [3], which makes it possible for users to implement their own models through FORTRAN code to simulate hybrid grids. However, this solution models power electronics-based resources as ideal power sources and its dependency on FORTRAN might be an obstacle for new users.

More recently, Modelica-based simulators have been gaining interest in the power system community. Modelica is a multiphysics object-oriented programming language, which is well suited for these types of applications. The Open-Instance Power System Library (Open-IPSL) for Modelica [4], [5], e.g., has been developed for years to allow users to simulate power systems with dynamic synchronous generator models and transmission grid modeled using phasor approximations and differential algebraic equations. This library has also been used to study integration of power electronics-based sources for large grid stability studies [6]. Complementary, another solution, which combines C++ and Modelica [7], has been proposed to lighten the computation costs when simulating large power systems. In parallel, electromagnetic transient (EMT-)type modeling tools based on the Modelica language have also emerged. In these EMT simulators, such as MSEMT [8], transmission line dynamics are no longer ignored, compared to phasor models.

Even if Modelica and FORTRAN are powerful programming languages, other languages remain more accessible. Python, for instance, is a popular language in multiple science and engineering fields, with various open-source libraries available. For example, the PyPSA project [9] made use of Python for power system analysis, although its main aim is to conduct grid-level techno-economic analyses. Another option, DPsim, a real-time simulator [10], makes use of a Python interface to simulate either dynamic phasor or EMT grid models in real time. However, it appears to be less suitable for converter control development.

All these various software projects are valuable assets for the electric power community. However, they do not allow researchers and engineers to develop novel methods at the power-conversion level since they are more adapted to large-scale validation of existing methods. The libraries presented in this paper focus on converter-level controllers and more specifically grid following (GFL) and grid forming (GFM) control structures rather than on their interactions with complex grids. The main motivation for this project, named *gritulator*, is to have a simple, reliable simulator based on Python for converter control research and education. Besides, it is also be possible to combine *gritulator* with the open-access scientific libraries available in Python.

The contributions of this paper can be listed as follows:

- An open-source Python-based simulator is developed, whose software structure is based on an ongoing project for motor drive simulation [11]. The present paper extends the motor drive libraries with grid converter ones.
- The implemented continuous-time models and discrete-time control methods are detailed in order to help first-time users.
- Implemented examples of already published control methods are presented and compared to experimental results obtained with a 12.5-kVA industrial converter.

In *gritulator*, programming skills in Python suffice to simulate test cases or to add new features to the existing control methods or continuous-time models. This makes it an accessible

This work was supported in part by the Research Council of Finland (352955) and in part by ABB Oy.

979-8-3503-9678-2/23/$31.00 ©2023 IEEE

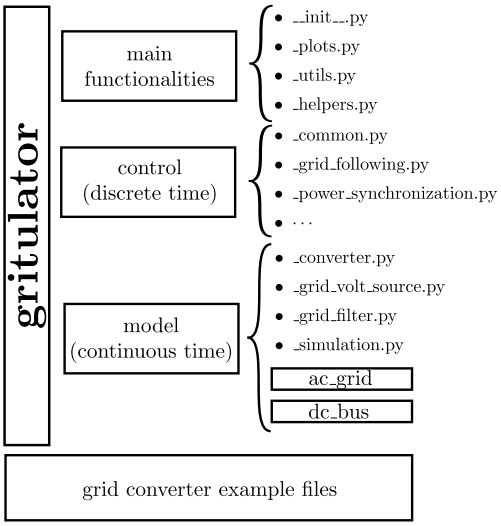


Fig. 1. Software architecture of *gritulator*.

and easy-to-use simulator for research and education, publicly available online [12].

II. Simulator Framework

A. Overall Structure

gritulator is originally a fork of the motor drive simulator *motulator* [13]. Some models and methods are shared between the two simulators and redundancies are avoided [1]. Hence, new modules can be easily shared between the two libraries. The overall structure of *gritulator* is given in Fig. 1. *gritulator*, similarly to *motulator*, is based on three main file categories:

- Main functionalities: including the scripts used to call the different libraries, simulate the system and plot the results in Python.
- Control: including all the control algorithms developed for both drive and grid applications (in discrete-time domain) and a module for the common shared functions.
- Model: including all the available models implemented in the continuous-time domain.

The following subsections describe different aspects of this simulator.

B. Interfaces and a Solver

Fig. 2 shows a generic block diagram of a grid converter system. The controller is usually a digital system, i.e., implemented in discrete time, while the plant is a continuous-time system.

The interface between the continuous-time and discrete-time systems is ensured by the simulation class. The set of differential equations is solved using an open-source explicit Runge-Kutta algorithm from the SciPy library [11].

C. Available Plant Models

The available modules used to build the plant model are highlighted in Fig. 3. This list of models from the library are

[1]Readers are invited to visit https://github.com/Aalto-Electric-Drives/gritulator to test the simulation platform with its most recent updates.

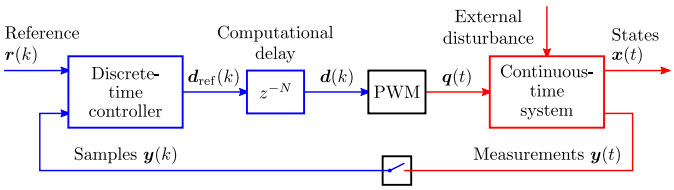


Fig. 2. Block diagram of a sampled-data system. Discrete-time elements are in blue, and continuous-time elements are in red [13].

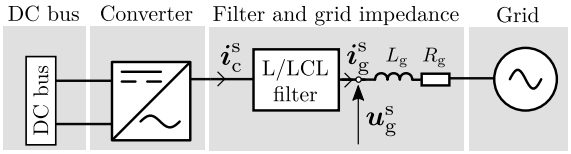


Fig. 3. Electrical diagram emphasizing the modules of *gritulator*. The modules used in *gritulator* are represented by the shaded boxes.

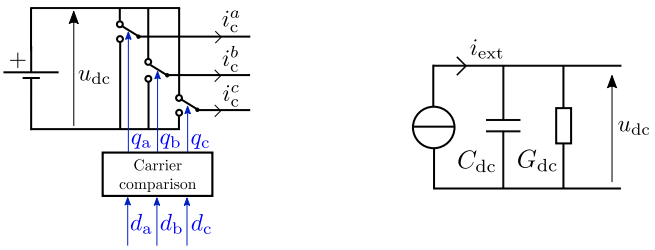


Fig. 4. Diagrams of the (a) AC-DC converter with the carrier comparison and the (b) DC-bus dynamic model.

summarized in the following paragraphs.

Fig. 4(a) shows the model of a two-level converter. A model with a carrier comparison is available in *gritulator*, where the states of the ideal switches are computed explicitly and provided to the solver, using a carrier comparison. If the switching ripples are not of interest, the user can also use a switching-cycle-averaged model which computes the duty ratios directly to switching states. More details about the converter models can be found in [11] as these models are shared between grid and drive applications.

The DC-bus model used by default is also described in Fig. 4(a) and relies on a voltage-source model. If the DC bus voltage dynamics are a matter of interest, a first-order dynamic model can be used, see Fig. 4(b). It corresponds to the following dynamic equation

$$\frac{\mathrm{d}u_{\mathrm{dc}}}{\mathrm{d}t} = \frac{1}{C_{\mathrm{dc}}}\left(i_{\mathrm{ext}} - i_{\mathrm{dc}} - G_{\mathrm{dc}}u_{\mathrm{dc}}\right) \tag{1}$$

where u_{dc} is the DC voltage, i_{ext} is the current supplied by an external source, i_{dc} is the DC-current pulled by the converter, C_{dc} is the DC-capacitance, and G_{dc} is the DC-conductance.

The AC-side impedance between the grid and the converter is described in the grid filter module. This module combines electrical dynamics of the inductive (L) or inductive-capacitive-inductive (LCL) filter of the converter main circuit and inductive-resistive grid impedance. The third-order

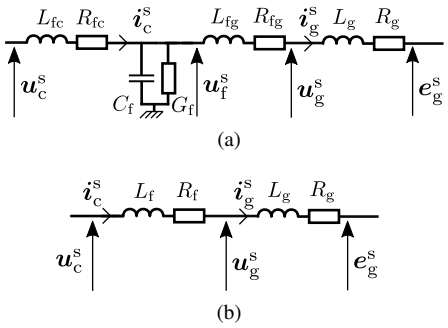


Fig. 5. Diagram of the filter and inductive-resistive grid impedance: (a) LCL filter; (b) L filter.

dynamic model of an LCL filter with the grid impedance, as implemented in *gritulator*, is shown in Fig. 5(a). The corresponding differential equations are

$$\frac{\mathrm{d}\boldsymbol{i}_{\mathrm{c}}^{\mathrm{s}}}{\mathrm{d}t} = \frac{1}{L_{\mathrm{fc}}}\left(\boldsymbol{u}_{\mathrm{c}}^{\mathrm{s}} - \boldsymbol{u}_{\mathrm{f}}^{\mathrm{s}} - R_{\mathrm{fc}}\boldsymbol{i}_{\mathrm{c}}^{\mathrm{s}}\right) \tag{2}$$

$$\frac{\mathrm{d}\boldsymbol{u}_{\mathrm{f}}^{\mathrm{s}}}{\mathrm{d}t} = \frac{1}{C_{\mathrm{f}}}\left(\boldsymbol{i}_{\mathrm{c}}^{\mathrm{s}} - \boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}} - G_{\mathrm{f}}\boldsymbol{u}_{\mathrm{f}}^{\mathrm{s}}\right) \tag{3}$$

$$\frac{\mathrm{d}\boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}}{\mathrm{d}t} = \frac{1}{L_{\mathrm{t}}}\left(\boldsymbol{u}_{\mathrm{f}}^{\mathrm{s}} - \boldsymbol{e}_{\mathrm{g}}^{\mathrm{s}} - R_{\mathrm{t}}\boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}\right) \tag{4}$$

where $\boldsymbol{i}_{\mathrm{c}}^{\mathrm{s}}$ is the converter-side current, $\boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}$ is the grid-side current, $\boldsymbol{u}_{\mathrm{c}}^{\mathrm{s}}$ is the converter output voltage, $\boldsymbol{u}_{\mathrm{f}}^{\mathrm{s}}$ is the LCL-filter internal voltage, $\boldsymbol{u}_{\mathrm{g}}^{\mathrm{s}}$ is the voltage at the point of common coupling (PCC) and $\boldsymbol{e}_{\mathrm{g}}^{\mathrm{s}}$ is the grid voltage. Peak-valued complex space vectors denoted with boldface are used for the AC-side variables, while real valued scalars are used for the DC-bus and mechanical variables. The AC-side variables are defined in the $\alpha\beta$ stationary frame. For the parameters, $\{L_{\mathrm{fc}}$,$R_{\mathrm{fc}}\}$ and $\{L_{\mathrm{fg}}$,$R_{\mathrm{fg}}\}$ are the converter-side and grid-side inductance and resistance terms respectively. The LCL-filter capacitance and conductance are defined by C_{f} and G_{f} respectively. $L_{\mathrm{t}} = L_{\mathrm{fg}} + L_{\mathrm{g}}$ and $R_{\mathrm{t}} = R_{\mathrm{fg}} + R_{\mathrm{g}}$ are the total inductance and resistance which include the grid inductance L_{g} and resistance R_{g}. An L-filter model is also available and shown in Fig. 5(b). It corresponds to a simplification of the previous model such that

$$\frac{\mathrm{d}\boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}}{\mathrm{d}t} = \frac{1}{L_{\mathrm{t}}}\left(\boldsymbol{u}_{\mathrm{c}}^{\mathrm{s}} - \boldsymbol{e}_{\mathrm{g}}^{\mathrm{s}} - R_{\mathrm{t}}\boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}\right) \tag{5}$$

where $\boldsymbol{i}_{\mathrm{c}}^{\mathrm{s}} = \boldsymbol{i}_{\mathrm{g}}^{\mathrm{s}}$ in this case and $L_{\mathrm{t}} = L_{\mathrm{f}} + L_{\mathrm{g}}$ and $R_{\mathrm{t}} = R_{\mathrm{f}} + R_{\mathrm{g}}$ where the filter parameters are L_{f} and R_{f}.

For the grid voltage model, different classes can be used to generate $\boldsymbol{e}_{\mathrm{g}}^{\mathrm{s}}$. The first one is an ideal voltage-source model, where the voltage magnitude and frequency are given by the user. It is possible to simulate three-phase faults by defining a time-dependent grid voltage magnitude. In addition, the grid model can also take into account the frequency dynamics of a grid based on the aggregated model of a synchronous generator shown in Fig. 6. In order to interconnect these sub-models to build the plant model, an additional class is used to link the inputs and outputs and formulate the ordinary differential equation problem.

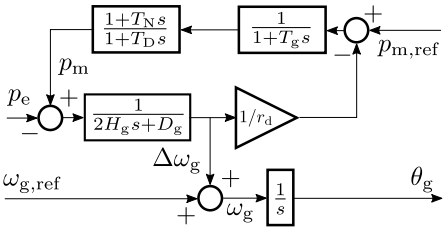


Fig. 6. Block diagram of the grid dynamic model with a mechanical model including the swing equation and TGOV1-type governor model [14].

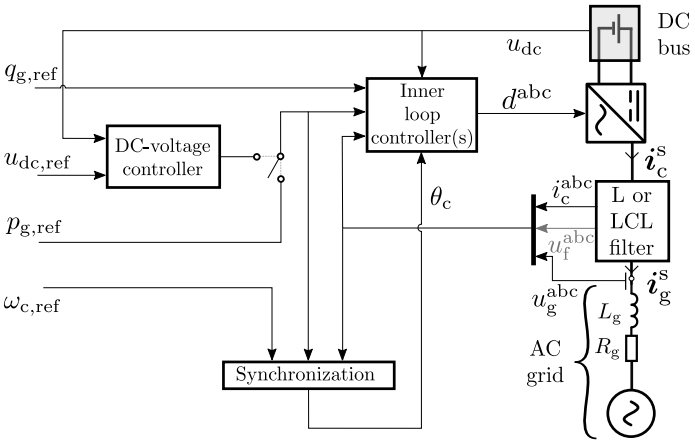


Fig. 7. Example control framework used for grid converters in *gritulator*.

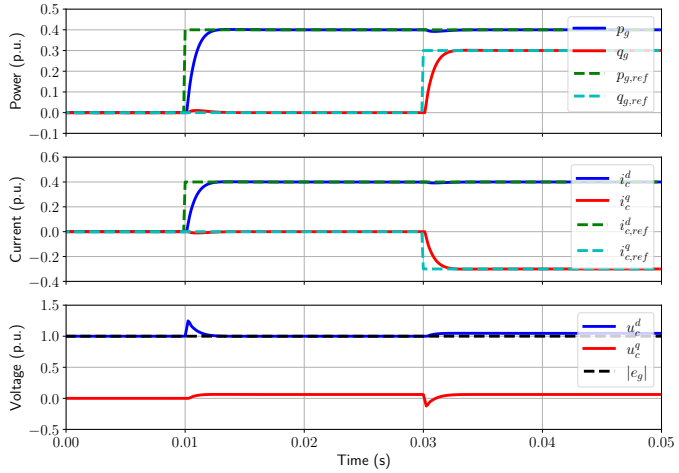
D. Available Control Methods

The control libraries already available in *gritulator* include grid-following [15] and grid-forming control [16] methods. In addition to active and reactive power control interfaces, the control system can easily be augmented with DC-voltage control [17]. Fig. 7 represents the overall control framework, which is given using general blocks to be compatible with the different methods. In the synchronization block, a phase-locked loop (PLL) or a power-synchronization law can be used, depending on the selected control method. The inner-loop controller can also contain different elements depending on the selected control method. To give an example, this controller can comprise a current-reference generator, current limitation, proportional-integral (PI) current controller and pulse-width modulation (PWM) when grid-following control is simulated.

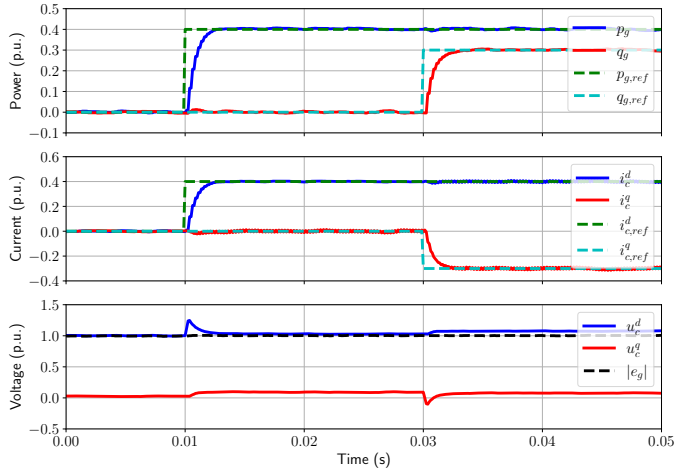
III. Examples

A. Context and Test Cases

In this section, two test cases are presented in order to compare the simulator with experimental test setup results. This test setup is not described in this paper but more details about it can be found in [18]. The switching frequency of the converter is selected to be 4 kHz and the sampling frequency of the discrete-time controller is 8 kHz. Active and reactive



(a)



(b)

Fig. 8. Response of a GFL-controlled converter with $L_g = 0$ mH with active and reactive power steps: (a) simulation results; (b) experimental results.

powers are plotted based on the measured PCC quantities using the following equations

$$p_g = \frac{3}{2}\,\mathrm{Re}\left\{\boldsymbol{u}_g^s \boldsymbol{i}_g^{s*}\right\} \qquad q_g = \frac{3}{2}\,\mathrm{Im}\left\{\boldsymbol{u}_g^s \boldsymbol{i}_g^{s*}\right\} \tag{6}$$

B. Grid-Following Control

A grid-following converter is simulated in order to study its power tracking capability, where both active and reactive power references are changed. In order to have a simple example, the grid is considered to be strong, i.e., there is no inductance between the PCC and the grid voltage ($L_g = 0$). The output filter is a simple L filter.

The results obtained with *gritulator* are given in Fig. 8(a) It can be seen that both active and reactive powers follow their references well, corresponding to the designed first-order response. The script used to obtain these results is given in Listing 1. Parameters were chosen in order to match the laboratory conditions. The experimental results are shown in Fig. 8(b) and match well the simulation results.

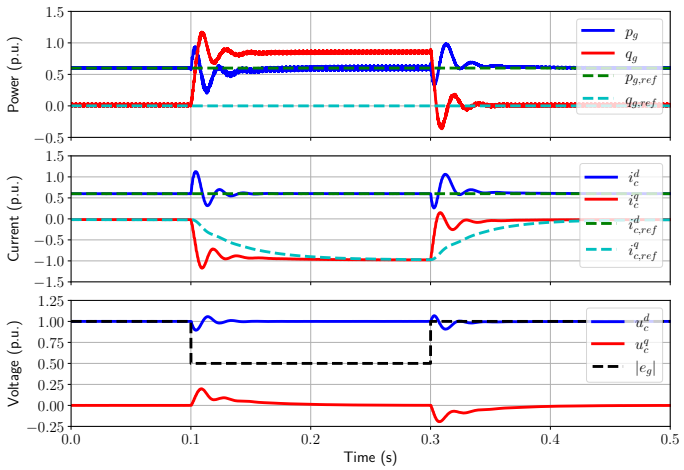
Code Listing 1. Script for simulating a 12.5-kVA GFL converter.

```
# Import the packages.
import numpy as np
from gritulator import model, control
from gritulator import BaseValuesElectrical
import plots_sim

# To check the computation time of the program
import time
start_time = time.time()

# Compute base values based on the nominal values.
base_values = BaseValuesElectrical(
U_nom=400, I_nom=18.04, f_nom=50.0, P_nom=12.5e3)

# Create the system model.
grid_filter = model.LFilter(L_f=6.3e-3, L_g=0, R_g=0)
grid_model = model.StiffSource(w_N=2*np.pi*50)
converter = model.Inverter(u_dc=653)
mdl = model.ac_grid.StiffSourceAndLFilterModel(
grid_filter, grid_model, converter)

# Control parameters
pars = control.grid_following.GridFollowingCtrlPars(
L_f=6.3e-3,
f_sw = 4e3,
T_s = 1/(8e3),
alpha_c = 2*np.pi*200,
i_max = 1.2*base_values.i,
)
ctrl = control.grid_following.GridFollowingCtrl(pars)

# Set the active and reactive power references.
ctrl.p_g_ref = lambda t: (t > .02)*(12.5e3)*0.4
ctrl.q_g_ref = lambda t: (t > .04)*(12.5e3)*0.3

# AC-grid voltage magnitude
e_g_abs_var =  lambda t: np.sqrt(2/3)*400
mdl.grid_model.e_g_abs = e_g_abs_var # grid voltage
   magnitude

# Create the simulation object and simulate it
sim = model.Simulation(mdl, ctrl, pwm=False)
sim.simulate(t_stop = .06)

# Plot results in SI or per unit values
plot_grid(sim, base=base_values,plot_pcc_voltage=True)
```

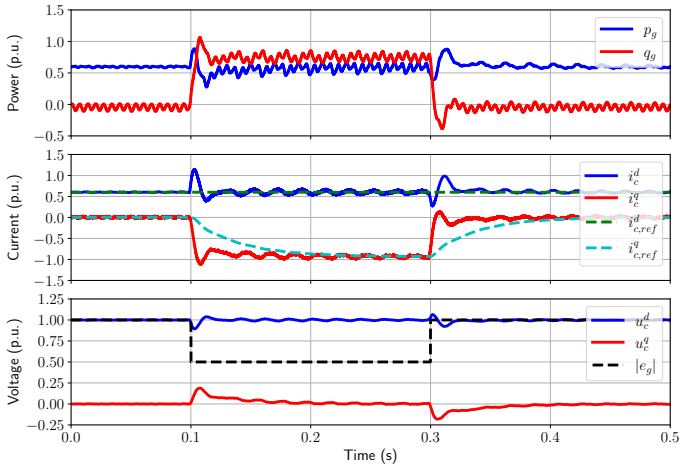
C. Reference-Feedforward PSC

In order to study the behavior of the converter in *gritulator* in grid-forming mode, reference-feedforward power synchronization control (RFPSC) [16] is also implemented and tested. In order to be complementary with the previous example, this one uses an LCL filter instead of an L filter. Furthermore, the inductance between the PCC and the grid is not zero anymore. The carrier comparison of the PWM is enable in *gritulator*. Moreover, this example shows how voltage sags can be simulated and how the converter responds in such cases.

Fig. 9(a) shows the simulation results, illustrating the voltage-source behavior of the converter when a -0.5-p.u. voltage sag occurs. The experimental results are shown in Fig. 9(b). The simulated and measured waveforms exhibit a similar behavior, in terms of pre-fault and post-fault operating points. In the experimental results, more high-frequency oscillations are observed during the sag when compared to the simulation results. If needed, nonidalities could also be modeled in *gritulator*.



(a)



(b)

Fig. 9. Response of a RFPSC-controlled converter with $L_g = 17.8$ mH and $R_g = 0.6\ \Omega$ following -0.5 p.u.-voltage sag: (a) simulation results; (b) experimental results.

D. Other Available Examples

In addition to the two presented examples, several other examples are available in GitHub [12]. These examples use, e.g., the DC-bus model shown in Fig. 4(b) and the AC electromechanical model shown in Fig. 6.

IV. Conclusions

An open-source simulator written in the Python language for studying grid converters is presented. The main structure of software and available libraries are introduced and some illustrative examples are given. The shown simulation examples are also compared with experimental test setup results. The simulator is a relevant tool for developing control algorithms. This solution could be used in academia for research and educational purposes as well as in industry for research and development.

In future work, more control methods could be added such as cascade controllers and other types of GFM strategies. In addition, more detailed AC grid models could be implemented, e.g., to describe more accurately low inertia power systems. DC-side models, including storage units or dynamic models of renewable sources, could also be relevant extensions.

References

[1] C. Breyer, S. Khalili, D. Bogdanov, M. Ram, A. S. Oyewo, A. Aghahosseini, A. Gulagi, A. A. Solomon, D. Keiner, G. Lopez, P. A. Ostergaard, H. Lund, B. V. Mathiesen, M. Z. Jacobson, M. Victoria, S. Teske, T. Pregger, V. Fthenakis, M. Raugei, H. Holttinen, U. Bardi, A. Hoekstra, and B. K. Sovacool, "On the history and future of 100% renewable energy Systems research," *IEEE Access*, vol. 10, pp. 78 176–78 218, 2022.

[2] F. Milano, L. Vanfretti, and J. C. Morataya, "An open source power system virtual laboratory: The PSAT case and experience," *IEEE Trans. Educ.*, vol. 51, no. 1, pp. 17–23, 2008.

[3] P. Aristidou and T. Van Cutsem, "Dynamic simulations of combined transmission and distribution systems using parallel processing techniques," in *Power Systems Computation Conference*, Wrocław, Poland, Aug. 2014.

[4] L. Vanfretti, T. Rabuzin, M. Baudette, and M. Murad, "iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations," *SoftwareX*, vol. 5, pp. 84–88, 2016.

[5] M. Baudette, M. Castro, T. Rabuzin, J. Lavenius, T. Bogodorova, and L. Vanfretti, "OpenIPSL: Open-instance power system library — update 1.5 to "iTesla power systems library (iPSL): A Modelica library for phasor time-domain simulations"," *SoftwareX*, vol. 7, pp. 34–36, Jan. 2018.

[6] J. C. Gonzalez-Torres, R. Mourouvin, K. Shinoda, A. Zama, and A. Benchaib, "A simplified approach to model grid-forming controlled MMCs in power system stability studies," in *IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, Espoo, Finland, Oct. 2021.

[7] A. Guironnet, M. Saugier, S. Petitrenaud, F. Xavier, and P. Panciatici, "Towards an open-source solution using modelica for time-domain simulation of power systems," in *IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Sarajevo, Bosnia and Herzegovina, Oct. 2018.

[8] A. Masoom, J. Mahseredjian, T. Ould-Bachir, and A. Guironnet, "MSEMT: An advanced Modelica library for power system electromagnetic transient studies," *IEEE Trans. Power Delivery*, vol. 37, no. 4, pp. 2453–2463, Aug. 2022.

[9] T. Brown, J. Hörsch, and D. Schlachtberger, "PyPSA: Python for power system analysis," *arXiv preprint arXiv:1707.09913*, 2017.

[10] M. Mirz, S. Vogel, G. Reinke, and A. Monti, "DPsim—A dynamic phasor real-time simulator for power systems," *SoftwareX*, vol. 10, p. 100253, Jul. 2019.

[11] L. Tiitinen, H. Hartikainen, L. Peretti, and M. Hinkkanen, "motulator: Motor drive simulator in Python," in *IEEE International Electric Machines & Drives Conference (IEMDC)*, San Francisco, CA, USA, May 2023.

[12] Aalto-Electric-Drives, "gritulator: Grid converter simulator in Python," 2023. [Online]. Available: https://github.com/Aalto-Electric-Drives/gritulator

[13] ——, "motulator: Motor drive simulator in Python," 2023. [Online]. Available: https://github.com/Aalto-Electric-Drives/motulator

[14] ENTSO-E, "Documentation on controller tests in test grid configurations," Tech. Rep., Nov. 2013.

[15] L. Harnefors and M. Bongiorno, "Current controller design for passivity of the input admittance," in *2009 13th European Conference on Power Electronics and Applications*, 2009.

[16] L. Harnefors, F. M. M. Rahman, M. Hinkkanen, and M. Routimo, "Reference-feedforward power-synchronization control," *IEEE Trans. Power Electron.*, vol. 35, no. 9, pp. 8878–8881, Sep. 2020.

[17] Namho Hur, Jinhwan Jung, and Kwanghee Nam, "A fast dynamic DC-link power-balancing scheme for a PWM converter-inverter system," *IEEE Trans. Ind. Electron.*, vol. 48, no. 4, pp. 794–803, Aug. 2001.

[18] R. Mourouvin, T. Nurminen, M. Hinkkanen, V. Pirsto, and J. Kukkola, "Multifunctional grid-forming cascade control for converters equipped with an LCL filter," in *IEEE 32nd International Symposium on Industrial Electronics (ISIE)*, Espoo, Finland, Jun. 2023.