A Short Tutorial On Using SGsim Framework For Smart Grid Applications

Abdalkarim Awad
Department of Electrical and Computer Engineering
Birzeit University, Palestine
akarim@ieee.org

Peter Bazan and Reinhard German
Computer Networks and Communication Systems
University of Erlangen, Germany
{peter.bazan, reinhard.german}@fau.de

ABSTRACT

Extensive usage of information and communication technology (ICT) characterizes the next generation power grid. ICT supports diverse smart grid applications from the level of energy management, all the way down to control and protection approaches. The evaluation of a smart grid application requires multidisciplinary tools that capture its versatile nature. In this paper, we present a short tutorial on the use of the open source co-simulation framework SGsim for smart grid applications. The framework combines two main simulators. On the one hand, it uses OMNeT++ to simulate data communication networks and to control the operation of the components. On the other hand, it uses OpenDSS to simulate the power grid.

1. INTRODUCTION

The enrichment of the power grid with ICT enables new applications which eventually will make it possible to run the power grid in an efficient, secure, reliable, sustainable, and economic way. For instance, applications such as Conservation Voltage Reduction (CVR) and Volt/VAR optimization have the potential to reduce the power consumption and optimize the operation of the power grid. The rapidly increasing penetration of fluctuating renewable energy sources (RES) brings new challenges to the power grid, especially inside the distribution network. For example, on a sunny day with high penetration of photostatic, the voltage can exceed the allowed limits (over-voltage) because of a low demand at midday. Therefore, it is important to coordinate the available components to avoid abnormal situations. In addition to the loads, a typical electricity distribution grid contains also components such as transformers and distributed energy resources. Additionally, energy storage elements will be part of future power electricity distribution grid. Connecting these components together through a data communication network is very crucial for the future power grid. It will make it possible to operate the power grid in an optimal way. Moreover, it will be possible to react very fast to emergency conditions. Smart grid applications include [5]:

- Volt and VAR control (VVC): Control of voltage regulation devices and reactive power compensation devices for the purpose of reducing consumer demand and energy loss, and maintain voltages at various points in the distribution network within acceptable limits.
- Fault detection, isolation and restoration (FDIR): This application aims at automating the field switching devices. It will use microprocessor based intelligent electronic devices (IEDs) and high-speed communication connections to enhance the reliability of the system and reducing the restoration time.
- Demand response (DR) management: DR can be used to shave the peak at high demand periods. This can be achieved by introducing a variable price in which the electricity will be expensive at high demand periods and cheap at low demand periods.
- Distributed energy resources (DER) integration and management: DER provides energy inside its local vicinity. With high penetration of DER, problems such as over-voltage appear and therefore suitable solutions should be provided. This application aims at improving the integration of DER in the power grid.
- Wide area monitoring, protection and control (WAMPC): This application extends the time resolution of traditional monitoring systems down to the sub-second time scale, making it capable of monitoring and reacting to dynamic instabilities in the grid.

Because testing new approaches and algorithms inside real electricity networks is difficult and sometimes impossible, simulation plays a key role to explore the benefits and risks of applying new smart grid applications. Through simulation, it is possible to perform comprehensive simulation experiments for different scenarios and approaches and this will eventually provide answers for specific questions. It is extremely difficult to seize the many-sided nature of smart grid in one tool. Through co-simulation it is possible to capture and evaluate different aspects of smart grids.

In this paper we provide a tutorial on SGsim. The co-simulation framework SGsim [3,4] uses the open source power simulator OpenDSS [1] to simulate electric power networks and it uses OMNeT++ [21] to simulate data communication networks as well as to build several components.

VALUETOOLS 2016, October 25-28, Taormina, Italy

DOI 10.4108/eai.25-10-2016.2267055

2. RELATED WORK

Smart grids are complex systems over multiple domains and for a number of these domains there already exist appropriate simulation tools. The coupling of such tools to a tailor-made smart grid co-simulation is part of the ongoing work. SGsim couples an electric power network simulator with a data communication network simulator and has two open source optimization tools integrated together with an interface to attach real phasor data concentrators. Other co-simulation frameworks are focused on different combinations of domains.

One example of a smart grid simulation tool used for co-simulation is GridLAB-D [7]. It is an agent-based hybrid simulation tool using continuous simulation for the energy part of the model and discrete event simulation for the communication part. There are no standard communication protocols implemented for the communication part. Only simplified network properties can be used such as bandwidth, latency, buffer size, and congestion.

GridLAB-D is used by the simulation platform GridSpice, together with MatPOWER and Amazon Web Services, for the distributed simulation of smart grids [2]. The integration of GridLAB-D in a co-simulation environment based on open source software is presented in [20]. Physical components like batteries are modeled with OpenModelica, the power system analysis is realized with the open source tool PSAT. In [14], the authors have used the commercial tool PowerFactory instead of PSAT and in [19] the co-simulation environment consists of PowerFactory for the power system, OMNeT++ for the communication infrastructure, and 4DIAC for the control system. In all three cases they investigated the smart charging of electric vehicles.

The focus of the co-simulation presented in [13] is on the coupling of the power distribution network simulator OpenDSS with the communication network simulator OMNeT++. This work is extended in [12] by a preliminary implemented information technology simulator. In general the authors suggest the use of tools like the multi-agent smart grid simulation platform MASGriP.

The MOSAIK framework controls the creation of the simulation models and the data flow between different simulators [11,16]. The simulation models are agent-based and the simulation execution is event-based with the possibility to manage different step sizes for the simulators. The framework provides an application programming interface for controlling commercial and open source simulation software. Simulators with a Functional Mock-up Interface (FMI) as well as those with a proprietary interface can be integrated. In the past, there was a lack of integrated communication network protocol simulators in the MOSAIK co-simulation framework. Therefore, the authors suggest in [8] a preliminary system architecture of integrating OMNeT++.

The co-simulation tool presented in [17] uses OPNET instead of OMNeT++ for the communication part and MATLAB/Simulink for the power system. The tool is then used for the detection of cyber-intrusion into the information infrastructure. Also, this tool provides only a part of the functionality of SGsim.

3. SGSIM

In this section we introduce SGsim framework. At the beginning we provide a detailed description of the simulator.

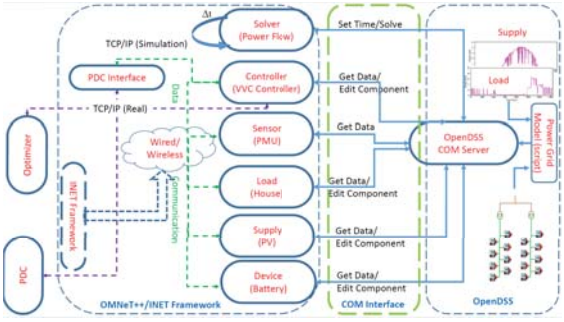


Figure 1: Structure of the SGsim framework with the connections between the components

Then we list the steps in order to install the software. At the end of this section we provide a simple example.

3.1 Description

SGsim [3, 4] is a co-simulation framework, which is based on two main simulators OpenDSS [9] and OMNeT++ [21]. OpenDSS provides an in-process Component Object Model (COM) server DLL designed to be driven from an external software. OMNeT++ is discrete event simulator which is used mainly as a data communication simulator. Additionally, several frameworks, such as the INET framework have been developed with well-tuned data communication components such as TCP/IP, 802.11, and Ethernet. Figure 1 shows the different components of the SGsim framework. Through the COM interface, the user is able to control the execution of the circuit and to change/add/remove different components. This is very helpful when simulating time-dependent scenarios. The main components of the simulator are:

- Power Grid Model: The OpenDSS is fed with a script that describes the components of the power grid and the interconnections. Time-dependent loads and supplies can be provided as text files. For household demand and photovoltaic supply real data from Pecan Street [15] will be used. This database provides a 1-min resolution aggregated power usage signal and the power consumption of individual devices. This can be very suitable in exploring applications such as DR.
- Solver: It controls the OpenDSS execution through the COM interface. It ensures time synchronization between OpenDSS and OMNeT++.
- Load: This is the OMNeT++ component of the load in the power grid, e.g., a house. It can measure power grid parameters such as voltage, current and power at a specific time through the COM interface. It is also possible to change load parameters, e.g., running time for Demand Response (DR) applications.
- Supply: Represents a power generation unit in OMNeT++, e.g., a DER. It is also possible to change supply parameters, e.g., regulate the output power (active and reactive power).
- Device: It represents power grid devices (e.g., battery, switch, capacitor bank, ...). Through the COM interface, it is possible to change the parameters, e.g., power factor.

- Sensor: It can only read data on a specific component (e.g., bus, load, DER) and send it to other components. For instance Phasor Measurement Unit (PMU) is considered as a sensor and it sends data to the Phasor Data Concentrator (PDC) Interface using simulated TCP/IP packets. The data is formatted using a standard (IEEE c37.118) so that the real PDC can interpret the packets.

- Controller: it represents an intelligent unit within the system. It receives data from other components and then, based on specific algorithms it can adapt system parameters. For instance, a CVR controller can change the voltage settings of a Load Tap Changer (LTC) in order to change the voltage of the transformer. In the SGsim framework, we integrated the open source optimization tools NLopt [10] and lpSolve [6]. It is possible also to call an external optimizer through a TCP/IP connection.

- PDC interface: It receives simulated packets inside the simulator and forwards it to real software components such as OpenPDC. In this case, the simulation should be run using the real-time mode.

We have implemented a DLL library to access the elements through the COM interface. The library provides the following functions:

- `run_command(char * command)` makes it possible to run a command. For instance, the command `edit load.LD1 kw=0.329 kvar=0` will set the active and reactive power values at LD1.

- `get_values_active_object(char * activeobject)` returns back different electricity measurements (e.g. voltage, current and active, and reactive Power) on a specific element.
  E.g., `values= fun_ptr(Load.LD1);` returns back the measurements at load LD1. `fun_ptr` is a pointer to the function `get_values_active_object`.

- `get_total_losses()` returns back the total losses in the network.

- `get_energy_active_object(char * Battery)` returns back the energy of a storage element (Battery).

OMNeT++ is a discrete event simulation and consequently, SGsim is a discrete event simulator in which OpenDSS is called periodically to solve the power flow in the electricity network.

3.2 Installation

The program is tested under a windows environment (Windows 7)

- OpenDSS: Install OpenDSS. Make sure the the `OpenDSSEngine.DLL` is registered. Usually it is installed automatically if you use the installer. You can manually register the server
  - run cmd to open a command prompt
  - go to the folder with the files
  - run the command `Regsvr32 OpenDSSEngine.DLL`

- OMNeT++: Install OMNeT++ (tested with version 4.4).

- INET Framework: Install the INET framework (tested with version 2.3.0)

- `SGsimLib.dll`: Store this file and give the full path in omnetpp.ini, for example:

  ```
  C:\\Users\\user1\\workspace
  \\SGsim\\simulations
  \\example1\\SGsimLib.dll
  ```

- Store the DSS file in the same folder!

- Provide the full path of the DSS file in omnetpp.ini file

  Important: you have to install Visual Studio or make sure you have the following DLL files: {msvcp120d.dll, msvcr100d.dll, msvcr120d.dll}.

3.3 Simple Example

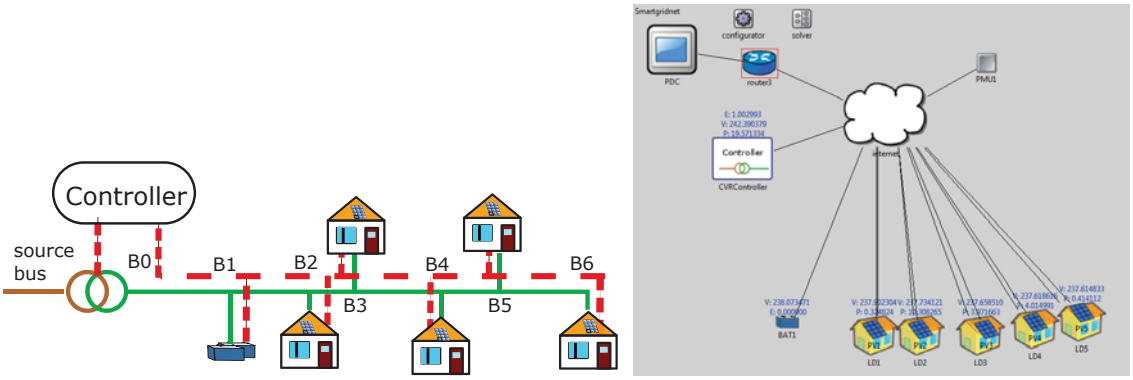
Figure 2 depicts a simple network that consists of 5 houses equipped with photovoltaic systems , and a storage element. At the beginning we have to provide a script that describes the topology of the network. The values of the parameters are not important because we can change the values from OMNeT++ during the run time. Listing 1 shows an OpenDSS script for the simple network. The script should start with `clear` so that there is no other networks in the memory. Then we create a voltage source (Line 3). After that we define the transformer and its terminals. We add here a controller for the transformer so that we can change the output voltage of the transformer. The parameters of the transmission line can be defined with `Linecode`. Then we define the connection between the buses. In OpenDSS, there are several models to represent loads. We used model 8 to represent the load which is suitable to study applications such as CVR. This load model use ZIP coefficients to represent loads as in Equations 1 and 2. The photovoltaic is defined with the keyword `PVsystem`. Additional parameters such as the relation between the efficiency and temperature can be included. The battery can be defined with the key word `storage`. Through OMNeT++ we can change the values of the parameters of a specific component, add new components or remove components.

$$P_{Li} = P_{0i}\left[Z_P\left(\frac{v_i}{v_0}\right)^2 + I_P\left(\frac{v_i}{v_0}\right) + P_P\right] \quad (1)$$

$$Q_{Li} = Q_{0i}\left[Z_q\left(\frac{v_i}{v_0}\right)^2 + I_q\left(\frac{v_i}{v_0}\right) + P_q\right] \quad (2)$$

Listing 1: OpenDSS code of a simple network

```
1 clear
2 // VSOURCE
3 new circuit.example1 basekV=11 pu=1.00 angle=0
      frequency=50 phases=3
4 set defaultBasefrequency=50
5 // TRANSFORMERS
6 new transformer.TR1 phases=3 windings=2 buses=(
      sourcebus, B0) conns=(wye, wye) kvs=(11,
      0.4) kvas=(250, 250) taps=(1,1)
7 //TRANSFORMER CONTROLLER
```



(a) (b)

Figure 2: The topology of the electricity and communication networks

```
8  new regcontrol.TR1 transformer=TR1 winding=2
       vreg=(242) ptratio=(1) band=1
9  // LINECODE AND LINES
10 new Linecode.LCode nphases=3 R1=0.320 X1=0.075
       C1=0 units=km
11 new line.Line01 bus1=B0 bus2=B1 Linecode=LCode
       length=0.5 units=km
12 new line.Line12 bus1=B1 bus2=B2 Linecode=LCode
       length=0.02 units=km
13 //repeat this line for the lines which
14 //connect Buses B2, B3, B4, B5, B6
15 // LOADS
16 new load.LD1 bus1=B2 kV=0.4 kW=2 PF=1 Model=8
       ZIPV=[0.85 -1.12 1.27 10.96 -18.73 8.77
       0.8]
17 //repeat this line for LD2 to LD5
18 // PV
19 new PVsystem.PV1 bus1=B2 phases=3 kV=0.4 KVA=10
       irrad=0.0 Pmpp=10
20 //repeat for PV2 to PV5
21 // BATTERY
22 New Storage.BAT1 Phases=3 Bus1=B1 kV=0.4
       kWRated=15 kWhRated=100 %stored=0 pf=1
23 // P.U.
24 Set voltagebases="11 0.4"
25 Calcvoltagebases
```

INET framework has been used to build the data communication network. The network consists of a model that represents the internet. Each element is equipped with a NIC that supports Ethernet, WLAN and PPP protocols. The houses are connected using PPP protocols that represents DSL connections. The NED language which is used in OMNeT++ has been used to describe the network. The source files of the network can be found at the sourceforge page [18]. Figure 2(b) shows a screenshot of the OMNeT++ simulation environment. It is preferred to use the same names in both OMNeT++ and OpenDSS. This makes the access to the components and their parameters more convenient. Some measurements are shown above each component. For instance, the voltage at the transformer (V) equals 242 volt, Power (P) is about 19 kW and the energy consumption (E) from the start of simulation is about 1 kWh. Each component produces a text file that contains the voltage and power at each time step. Of course, it is possible to include other measurements in this file.

Table 1: Parameters

Parameter	Value
PV	10 kVA
Battery power	15 kVA
Battery capacity	100 kWh
v_{thr1}, v_{thr2} ,v_{thr3} ,v_{thr4}	249, 215, 220, 244 volt
r+xj	$(0.320 + 0.075j)/km$
l_1	500 m
l_2	20 m

4. CASE STUDY: VOLTAGE CONTROL OF ELECTRICITY DISTRIBUTION GRID

Algorithm 1 The house application

Require: Get Voltage (V_i) at load i
Ensure: Sending current voltage value
1: **if** $(V_i) \leq v_{thr1}$ **then**
2: send an under-voltage warning message to the controller
3: **else if** $(V_i) \geq v_{thr2}$ **then**
4: send an over-voltage warning message to the controller
5: **else**
6: send a normal-voltage periodic message
7: **end if**
8: repeat

Voltage control is an important task of the distribution grid automation. Traditionally voltage control is limited to the Tap Changer. The tap changer increases or decreases the winding ratio of the upper/lower voltage side of the transformer. The value of the transformer ratio is a task of the network planning and has to ensure that all customers all the time are served with a voltage within the allowed band. Therefore, the ratios have to be selected and fixed based on extreme load conditions and the voltage drop along the feeder. This type of planning is no longer suitable for the next generation power grid where bi-directional power flows occur. The voltage profile in a traditional power grid is shown in Figure 3(a). As can be seen, the voltage declines along the feeder. The voltage profile becomes more com-

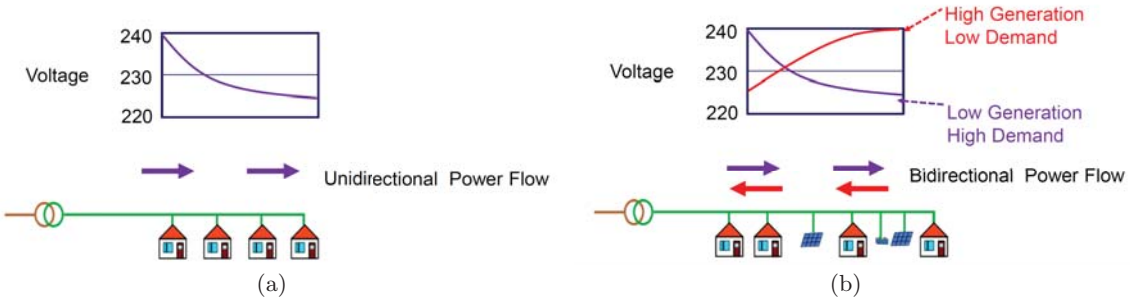


Figure 3: Voltage profiles in traditional networks (a) and in current and future networks (b)

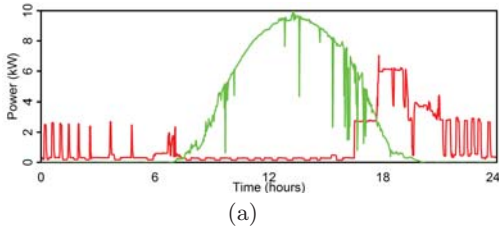


Figure 4: Demand (red) and supply (green) profiles

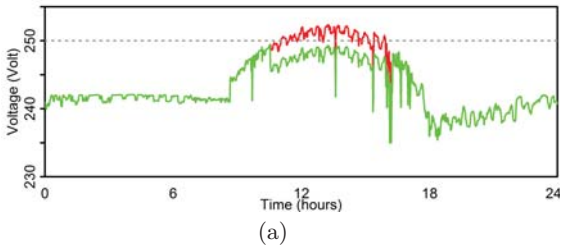


Figure 5: Voltage levels at the house at the end of the feeder without a battery (red) and with a battery (green)

plex as shown in Figure 3(b). The conditions are no longer strong/weak demand but instead are weak load/strong supply and strong supply/weak supply. New algorithms are required that are able to coordinate the operation of the different components on the power grid such as storage units, elastic loads, as well as transformers. For this purpose, the voltage measurements at end users is required. Traditionally, the voltage measurements at the customer side is not available. Here two approaches are possible: either installing measurement units or installing smart meters. Controlling the active/reactive power can be used to control the voltage at the customer side. This can be done by several methods. Introducing a variable tariff can increase the demand during the high power injection and reduce the demand during the low power injection. Another approach is to control the active/reactive power of distributed energy resources. Batteries can also be used to control the voltage at the customer side. They can be charged at high power injection periods and discharged at low power injection periods.

Algorithm 2 The controller application

Require: Receive warning voltage messages from houses
Ensure: Transformer Output Voltage v_{TR}

```
if Under − Voltage then
    send a DISCHARGE signal to the battery
    State=DISCHARGE
else if Over − Voltage then
    send a CHARGE signal to the battery
    State=CHARGE
else if (V_i ≥ v_thr3)&(State == DISCHARGE) then
    send an IDLE signal to the battery
else if (V_i ≤ v_thr4)&(State == CHARGE) then
    send an IDLE signal to the battery
end if
```

In this case study we use the network shown in Figure 2(a). The distance between the transformer and the first house (l_1) is 500 m and between two houses (l_2) 20 m. A community storage unit is located inside the network. We exploit the available storage element to decrease the voltage at the high supply/low demand period and increase the voltage at low supply/high demand period. We use power demand and supply profiles from the Pecan street database [15]. The red line in in Figure 4(a) depicts a typical household load and the green line depicts a typical supply. Every house sends periodically its measured voltage (Algorithm 1). This algorithm is implemented as C++ program inside the house component. These measurements can be used to implement smart grid applications such as CVR and VVC. Yet, if the voltage at a house is above or below specific limits, a warning message will be sent. Upon receiving these messages, the controller can take actions to prevent unacceptable voltage values. In this case study, the controller will send a charge command for an over-voltage warning message and a discharge command for a low-voltage warning message as shown in Algorithm 2, which is implemented inside the controller component. The implementation of the algorithms can be found at [18]. The discharging process will continue until the voltage is raised above a specific value (v_{thr3}) and the charging process will continue until the voltage is below a specific value (v_{thr4}). Table 1 summarizes the simulation parameters. The parameters can be set in the omnetpp.ini configuration file.

Figure 5(a) shows the voltage at the last house at the end of feeder with and without using the battery units. As can be seen, the voltage raises during the midday to high levels (red line). This occurs because of the high electricity injection

from the solar panels and the low demand. Exploiting the available storage system reduces the voltage to acceptable limits (green line).

The approach is based on simple rules to react to abnormal voltage levels. A more complex approach which can employ optimization, can be used. For instance, it is possible to use demand response or control the distributed energy resources to maintain the voltage within acceptable limits as well as to improve the operation of the system.

5. CONCLUSION

In this paper we presented a tutorial on SGsim that enables the investigation of smart grid applications inside power distribution networks. SGsim is a framework that consists of the data communication simulator OMNeT++ and the power grid OpenDSS. This way, it is possible to investigate the impact of data communication systems on the electricity networks. Through a case study with real data, we have shown the possibility to mitigate over-voltage through controlling the storage units.

Acknowledgment

Peter Bazan is also a member of "Energie Campus Nürnberg", Fürther Str. 250, 90429 Nürnberg. His research was performed as part of the "Energie Campus Nürnberg" and supported by funding through the "Aufbruch Bayern (Bavaria on the move)" initiative of the state of Bavaria.

6. REFERENCES

[1] EPRI Electrical Power Research Institute. http://sourceforge.net/projects/electricdss/, 2016. [Online; accessed 20-July-2016].

[2] K. Anderson, J. Du, A. Narayan, and A. E. Gamal. GridSpice: A distributed simulation platform for the smart grid. *IEEE Transactions on Industrial Informatics*, 10(4):2354–2363, Nov. 2014.

[3] A. Awad, P. Bazan, and R. German. SGsim: a Simulation Framework for Smart Grid Applications. In IEEE, editor, *Proceedings of the IEEE International Energy Conference (ENERGYCON 2014)*, pages 730–736, Dubrovnik, Croatia, May 2014.

[4] A. Awad, P. Bazan, and R. German. SGsim: Co-Simulation Framework for ICT-Enabled Power Distribution Grids. In *18th International GI/ITG Conference on Measurment, Modelling and Evaluation of Computing Systems and Dependability and Fault-Tolerance (MMB & DFT 2016)*, Münster, Germany, April 2016.

[5] L. T. Berger and K. Iniewski, editors. *Smart Grid Applications, Communications and Security*. John Wiley & Sons, 2012.

[6] M. Berkelaar et al. lpSolve: interface to *Lp_solve* v. 5.5 to solve linear/integer programs. http://cran.r-project.org/web/packages/lpSolve/lpSolve.pdf, 2015. [Online; accessed 20-July-2016].

[7] D. P. Chassin, J. C. Fuller, and N. Djilali. GridLAB-d: An agent-based simulation framework for smart grids. *Journal of Applied Mathematics*, 2014:1–12, 2014.

[8] J. Dede, K. Kuladinithi, A. Förster, O. Nannen, and S. Lehnhoff. Omnet++ and mosaik: Enabling simulation of smart grid communications. *arXiv preprint arXiv:1509.03067*, 2015.

[9] EPRI Electrical Power Research Institute. Home page, Oct. 2016.

[10] S. G. Johnson. The NLopt nonlinear-optimization package. http://ab-initio.mit.edu/nlopt, 2016. [Online; accessed 20-July-2016].

[11] S. Lehnhoff, O. Nannen, S. Rohjans, F. Schlogl, S. Dalhues, L. Robitzky, U. Hager, and C. Rehtanz. Exchangeability of power flow simulators in smart grid co-simulations with mosaik. In *2015 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, pages 1–6. Institute of Electrical & Electronics Engineers (IEEE), Apr. 2015.

[12] M. Lévesque, C. Béchet, E. Suignard, M. Maier, A. Picault, and G. Joos. From co- toward multi-simulation of smart grids based on HLA and FMI standards. *The Computing Research Repository (CoRR)*, 2014.

[13] M. Lévesque, D. Q. Xu, G. Joós, and M. Maier. Communications and power distribution network co-simulation for multidisciplinary smart grid experimentations. In *Proceedings of the 45th Annual Simulation Symposium*, page 2. Society for Computer Simulation International, 2012.

[14] P. Palensky, E. Widl, M. Stifter, and A. Elsheikh. Modeling intelligent energy systems: Co-simulation platform for validating flexible-demand EV charging management. In *2014 IEEE PES General Meeting | Conference & Exposition*, page 1. Institute of Electrical & Electronics Engineers (IEEE), July 2014.

[15] Pecan street database. Home page, Oct. 2016.

[16] S. Rohjans, S. Lehnhoff, S. Schutte, S. Scherfke, and S. Hussain. mosaik-a modular platform for the evaluation of agent-based smart grid control. In *IEEE Innovative Smart Grid Technologies Europe (ISGT EUROPE)*, pages 1–5, 2013.

[17] M. A. H. Sadi, M. H. Ali, D. Dasgupta, and R. K. Abercrombie. OPNET/simulink based testbed for disturbance detection in the smart grid. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference on - CISR 15*, page 17. ACM, Association for Computing Machinery (ACM), 2015.

[18] SGsim. Home page, April 2016. https://sourceforge.net/projects/sgsim/.

[19] M. Stifter, J. H. Kazmi, F. Andren, and T. Strasser. Co-simulation of power systems, communication and controls. In *2014 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, pages 1–6. Institute of Electrical & Electronics Engineers (IEEE), Apr. 2014.

[20] M. Stifter, E. Widl, F. Andrén, A. Elsheikh, T. Strasser, and P. Palensky. Co-simulation of components, controls and power systems based on open source software. In *2013 IEEE Power & Energy Society General Meeting*, pages 1–5. IEEE, Institute of Electrical & Electronics Engineers (IEEE), 2013.

[21] A. Varga. The OMNeT++ Discrete Event Simulation System. In *European Simulation Multiconference (ESM 2001)*, Prague, Czech Republic, June 2001.