MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Power System Modeling for Identification and Control Applications using Modelica and OpenIPSL

Vanfretti, Luigi; Laughman, Christopher R.

TR2024-112 August 22, 2024

Abstract

The open-access Modelica language offers unique modeling features that enable power system modeling for identification and control applications. In this paper, we illustrate how the language can be used for these purposes leveraging the Modelica-based OpenIPSL library. With the goal of expanding the application of the language in system identification and control applications, the paper provides open-source models of different networks that can be re-utilized by the control community. Although the models in this paper target specific power system dynamic phenomena, that is, local low-frequency oscillation monitoring and control, they can easily be extended to address other power system dynamic issues.

Conference on Control Technology and Applications (CCTA) 2024

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Power System Modeling for Identification and Control Applications using Modelica and OpenIPSL

Luigi Vanfretti[†] and Christopher R. Laughman[‡]

***Abstract*— The open-access Modelica language offers unique modeling features that enable power system modeling for identification and control applications. In this paper, we illustrate how the language can be used for these purposes leveraging the Modelica-based OpenIPSL library. With the goal of expanding the application of the language in system identification and control applications, the paper provides open-source models of different networks that can be re-utilized by the control community. Although the models in this paper target specific power system dynamic phenomena, that is, local low-frequency oscillation monitoring and control, they can easily be extended to address other power system dynamic issues.**

Modelica, OpenIPSL, Power Systems, Power Grid, Identification, Oscillation Monitoring, PSS

I. Introduction

Power systems represent complex critical infrastructure for which both physics- and data-based models are crucial for planning and operation. The operation of these systems involves monitoring tasks that exploit system identification techniques, and planning involves control design tasks on different spatio-temporal scales (as shown in Fig. 1). A loose taxonomy of the phenomena categorizes different system dynamics as electromagnetic transients (EMT) at the sub-millisecond range, electromechanical transients (TS) within the millisecond to tens of minutes range, and quasi-steady state (QSS) for phenomena lasting minutes to hours. The challenging nature of these monitoring, planning, and control problems has resulted in a wide range of modeling and simulation efforts dating back to the 1970s [1]. These efforts initially produced software related to specific time scales and system size [2], [3], which have been extended over the course of the intervening years to domain-specific software tools [4], [5].

New opportunities related to the integration of electronic-based renewable energy sources into existing power systems have motivated the adoption of object-oriented modeling languages and tools to exploit the advances that these languages and environments offer beyond domain-specific tools [6], [7]. Unlike the domain-specific tools of the power system community, this approach requires the means to automatically and symbolically manipulate heterogeneous equation systems of differential algebraic equations (DAEs). One of the results of these efforts to expand system modeling to heterogeneous systems has been the development of the open standard Modelica language, which enables the use of modern modeling concepts while providing interoperability among multiple software tools. These capabilities make Modelica attractive for modeling power systems, with substantial benefits over domain-specific and general-purpose computing tools [8].

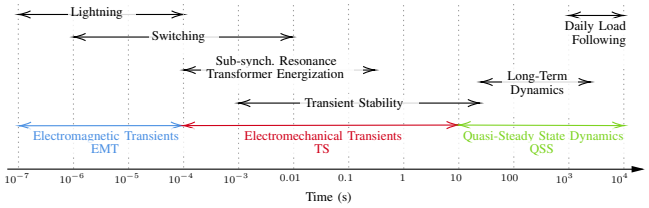

Fig. 1. Power system phenomena across multiple time-scales.

The use of Modelica for power system modeling was first reported in 2000 [9], which showed the possibilities of using Modelica to address the phenomena of the power system on all time scales in Fig. 1 and prompted a host of follow-up efforts. Winkler [10] surveyed open source libraries available up to 2017 covering both EMT and TS (see Fig. 1), while more recent industrial and academic efforts have included libraries and tools from RTE [11], as well as the `MSEMT` library for EMT modeling [12] and the `TransiEnt` library for integrating multiple energy domains, such as gas networks, with the grid [13].

In this paper, we present the use of a library for TS (also known as "phasor" modeling[1]) called `OpenIPSL` [14] to demonstrate how the Modelica language and the OpenIPSL library can be used for power system modeling to assist system identification and control applications. We provide sample models that can be reused and extended for the application of identification and control methods, and demonstrate the use of these methods on an example motivated by recent literature [15], [16]. Ultimately, we hope to expand the application of the Modelica language and the OpenIPSL library within the fields of system identification and control as a powerful alternative to domain-specific power system tools.

Notation and Resources: In this paper, the `typewriter` font is used to reference the syntax of the Modelica language within the text. It is also used to define the names of Modelica libraries, models, and variables that employ the dot notation. For example, the dot notation for the parameter `grid.line.R` indicates a model `grid` that contains a transmission line model instance `line` with a value for the resistance parameter named `R`. We also make frequent ref-

[†]LV is with the ECSE Department, Rensselaer Polytechnic Institute, Troy, NY, USA. vanfrl@rpi.edu

[‡]CR is with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA.laughman@merl.com

[1]Although OpenIPSL is designed for phasor modeling, it can also cover EMT and QSS dynamics (see Fig. 1) with extensions. For example, it can include EMT behavior [17] and long-term dynamics [18].

erence to a few key common Modelica resources, including the Modelica Standard Library (MSL) [19] and the Modelica language specification (MSPEC) [20].

II. Modeling

A. Phasors

AC power grids operate at synchronous frequencies, typically at a nominal value 50 or 60 Hz, by maintaining the frequency seen by the power generator's voltages and currents. The time scales in Fig. 1 thus characterize dynamics that are excited around this frequency. This is achieved in conventional synchronous generator-based power plants by controlling the rotor speed of the generator, while renewable energy sources modulate the electronic power converter that connects them to the grid [6].

The voltage ($\tilde{V}_e(t)$) and current ($\tilde{I}_e(t)$) signals in an AC power system can be represented via Euler's equations as

$$\begin{cases} \tilde{V}_e &= \sqrt{2}V_{RMS}e^{j\omega_n t+\theta} = \tilde{V}\sqrt{2}e^{j\omega_n t} \\ \tilde{I}_e &= \sqrt{2}I_{RMS}e^{j\omega_n t+\theta+\phi} = \tilde{I}\sqrt{2}e^{j\omega_n t}, \end{cases} \quad (1)$$

where $\omega_n = 2\pi f_n$ with f_n is the nominal frequency of the grid, θ is a phase displacement between the voltage wave and a reference, ϕ is a phase displacement between voltage and current waves, and $\tilde{V} = V_{RMS}e^{j\theta}$ and $\tilde{I} = I_{RMS}e^{j\theta+\phi}$ are the voltage and current phasors, respectively, where RMS indicates the root mean squared quantities.

By grouping the term $j\omega_n t$ that describes the synchronous frequency of the grid into a phasor representation, each electrical quantity can be represented by its RMS magnitude and the phase angle. This allows us to model the voltage and current at any node of the network as

$$\tilde{V} = v_r + jv_i, \text{ and } \tilde{I} = i_r + ji_i. \quad (2)$$

where v_r, v_i, i_r and i_i are the electrical quantities that will be used to define the basic interface used to couple different components in OpenIPSL.

B. Overview of the OpenIPSL Library

OpenIPSL is an open source, tool-agnostic Modelica library for phasor-based modeling and simulation for dynamic analysis of power systems. It is also capable of modeling distribution networks with unbalanced three-phase phasors [21] without requiring the use of cosimulation, as is the case for many domain-specific tools. In addition, if facilitates multi-time-scale modeling by integrating the EMT and TS models [17].

The main structure of the OpenIPSL library is shown in Fig. 2 (A). It is a Modelica `package`, which is conceptually organized like a directory to provide encapsulation and a high-level structure. While a variety of sub-packages are evident, the following are particularly relevant to this discussion:

- `OpenIPSL.Electrical` includes models of devices that are used to model an electrical power system. As shown in 2(B), it includes models of electrical machines, loads, transmission lines, transformers, and regulators/-control devices. Each of the sub-packages in Fig. 2 has its own sub-packages (e.g., Fig. 2(C)); for example, `Renewables` includes other sub-packages that can be used to model battery energy storage systems (BESS) and other renewable energy sources [6].

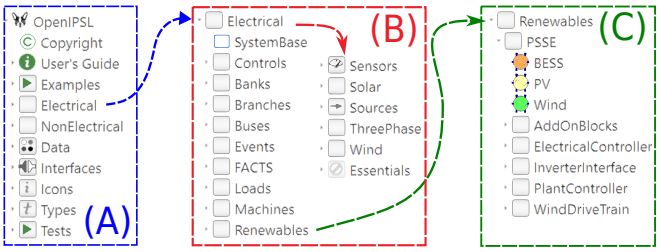

Fig. 2. OpenIPSL (v3.0.1-dev) Library Overview

- `OpenIPSL.NonElectrical` consists of specialized blocks, functions,and models that are used by the models of the `Electrical` package. It includes extensions of MSL components as well as specialized components that are necessary for the simulations to match those of PSS/E [4], [22].
- `OpenIPSL.Interfaces` is the sub-package from which all `Electrical` models inherit, including `connectors` and an a `partial model` for a generator. The `PwPin` Modelica connector and its variants define the *physical* phasor-based connector on which the library is based. The `Generator` partial model is an interface that can be used to build detailed power generation units with internal components.

C. Types and Connectors

The basic connector `PwPin` is used to provide an *acausal* interface between components of the `OpenIPSL.Electrical` package, the implementation of which is provided in Listing 1. This connector contains voltages (`vr` and `vi`, Lines 1, 2) that must satisfy equality constraints (e.g., KVL) across component boundaries, as well as currents (`ir` and `ii`, Lines 4, 5) that must satisfy sum-to-zero constraints (e.g., KCL) across component boundaries. The `flow` prefix indicates the presence of the sum-to-zero constraints to enable a Modelica tool to automatically generate the corresponding equations when connecting components. In addition, the types of these variables (e.g., `Types.PerUnit`, `SI.Voltage`, `Real`) are specified using physics-based units to enable unit checking by Modelica tools for model correctness.

Listing 1. Excerpt of the `PwPin` Connector in OpenIPSL.

```
1 connector PwPin
2   Types.PerUnit Real vr "Real part, voltage";
3   Types.PerUnit Real vi "Imaginary part, voltage";
4   flow Real ir(start=Modelica.Constants.eps) "Real part
        , current";
5   flow Real ii(start=Modelica.Constants.eps)         "
        Imaginary part, current";
6 end PwPin;
```

The instantiation of these variables also allows the specification of variable attributes, such as `start`. These attributes often provide valuable information that solver can use to improve the numerics of a particular problem. For example,

the `start` attribute can provide an initial guess to assist in the initialization of the DAEs resulting from a compiled model. Similarly, the `nominal` attribute can be used by Modelica tools to apply appropriate variable scaling, thus enabling the tool to determine tolerances that aid in avoiding round-off or truncation errors. The use of such attributes has significantly improved issues with the initialization and simulation of OpenIPSL models.

D. Object-Oriented Component Models

The `PwPin` connector is instantiated by all the power device models within the `OpenIPSL.Electrical` package to enable the interconnection of electrical components. We next present a synchronous generator model `Electrical.Machines.PSAT.Order2` that includes both algebraic and differential equations from the library to illustrate how object-oriented modeling is used to build component models. Several other models that have been validated against Siemens PTI PSSE® [4] are also available in OpenIPSL [6], [22].

A classical electromechanical model for a synchronous generator [23] was constructed neglecting electrical dynamics and making other simplifying assumptions, and its implementation was verified against PSAT [24]. The model is described by:

$$\begin{cases} \dot{\delta} & = \Omega_b(\omega - 1) \\ \dot{\omega} & = (p_m - p_e - D(\omega - 1))/M \end{cases} \tag{3}$$

where δ and ω are the machine's rotor angle and speed, Ω_b is the base frequency in rad/s, p_m the mechanical power, p_e the electrical power, D the damping coefficient, and $M = 2H$ where H is a lumped inertia of the turbine-generation system in sec. The electrical power output p_e is defined as:

$$p_e = (v_q + r_a i_q)i_q + (v_d + r_a i_d)i_d \tag{4}$$

where the subscripts dq denote the dq-axis voltages and currents, and r_a is the armature resistance. The dq voltages and currents arise from the application of Park's "two-reaction theory", which can be found in [23], and are linked to the power network at a bus by

$$v_d = v\sin(\delta - \theta) \;\text{ and }\; v_q = v\cos(\delta - \theta) \tag{5}$$

where v and θ are the bus voltage magnitude and angle at the bus. Finally, The dq-voltages and currents are related by

$$\begin{cases} 0 & = v_q + r_a i_q - e'_q + x'_d i_d \\ 0 & = v_d + r_a i_d - x'_d \end{cases} \tag{6}$$

where e'_q is constant, i.e., it neglects the electrical dynamics, and x'_d is the d-axis transient reactance of the machine.

One commonly used design pattern in Modelica makes extensive use of inheritance for model creation, as in this simple model of a synchronous machine. We first define a base model `pfComponent` that defines a set of parameters common to all components derived from power flow data, such as bus voltages or initial real and reactive power injections. The model `baseMachine`, an excerpt of which is shown in Listing 2, extends `pfComponent` with an acausal set of equations relating the terminal voltage and power, as well as the mechanical equations (see Lines 12-14 in Listing 2), which are common to many detailed machine models [23]. We can see the use of inheritance in Lines 2-4, which extend `pfComponent` and instantiate the connector `PwPin` to enable connections between the machine and other electrical components. Lines 9-10 & 17-18 are then used to relate the connector variables to the model equations. After declaring the parameters of (3)-(6), these equations are implemented on Lines 12-20, where the terms including `S_SBtoMB` and `I_MBtoSB` are used to change the base per unit value from machine to system base. Note that p_m is provided as an input to the model by instantiating the `RealInput` MSL `block` named `pm` in Line 7.

Listing 2. Excerpt of the `.PSAT.BaseClasses.baseMachine` model.

```
1  partial model baseMachine
2    extends OpenIPSL.Electrical.Essentials.pfComponent
         (...);
3    OpenIPSL.Interfaces.PwPin p(vr(start=vr0),vi(start=
         vi0), ir(start=ir0),ii(start=ii0));
4    parameter Types.Time M "Mechanical starting time, 2H
         [Ws/VA]";
5    ...
6    Modelica.Blocks.Interfaces.RealInput vf "Field
         voltage [pu]";
7    Modelica.Blocks.Interfaces.RealInput pm(start=pm00) "
         Mechanical power [pu]";
8  equation
9    v = sqrt(p.vr^2 + p.vi^2);
10   anglev = atan2(p.vi, p.vr);
11   der(delta) = w_b*(w - 1);
12   if D > Modelica.Constants.eps then
13     der(w) = (pm*S_SBtoMB - pe - D*(w - 1))/M;
14   else
15     der(w) = (pm*S_SBtoMB - pe)/M;
16   end if;
17   [p.ir; p.ii] = -[sin(delta), cos(delta); -cos(delta),
         sin(delta)]*[id; iq]*I_MBtoSB;
18   [p.vr; p.vi] = [sin(delta), cos(delta); -cos(delta),
         sin(delta)]*[vd; vq]*V_MBtoSB;
19   ...
20   pe = (vq + ra*iq)*iq + (vd + ra*id)*id;
21 end baseMachine;
```

Listing 3. Excerpt of `*.Electrical.Machines.PSAT.Order2`

```
1  model Order2 "Second Order Synchronous Machine ..."
2    extends BaseClasses.baseMachine(vf(start=vf00), xq0=
         x1d);
3  protected
4    parameter Real K=1/(ra^2 + x1d^2) "scaling const.";
5    parameter Real c1=ra*K "scaled ra";
6    parameter Real c2=x1d*K "scaled x'd";
7    parameter Real c3=x1d*K "scaled x'd";
8    parameter Types.PerUnit vf00=V_MBtoSB*(vq0 + ra*iq0 +
         x1d*id0) "Initial value (SB)";
9  equation
10   id = -c1*vd - c3*vq + vf_MB*c3;
11   iq = c2*vd - c1*vq + vf_MB*c1;
12   vf0 = vf00;
13 end Order2;
```

While the `baseMachine` model implements most of the equations for the machine, Equation (6) is not implemented because there are alternate formulations of the electrical dynamics of the machine, e.g., adding dq circuits and effects of speed variation on fluxes [23], [24]. We can therefore

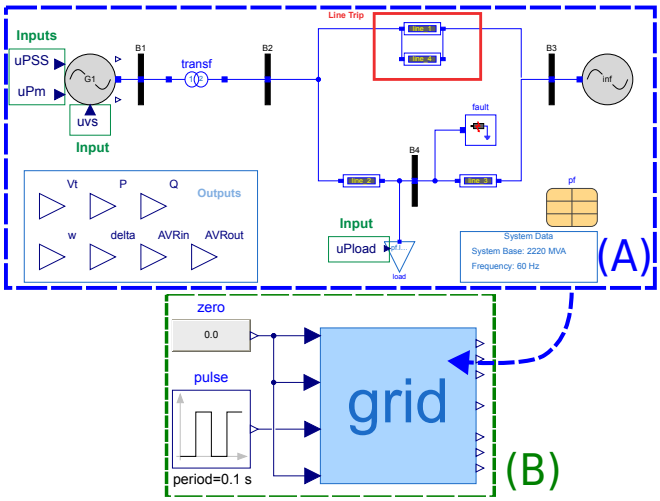

Fig. 3. `Example1.Base.Systems.gridIO` a SMIB Power System Model with I/O Interfaces: (a) model with I/O and (b) model used as a block.

extend the `baseMachine` model to a set of complete machine models, each of which implements a different model of the electrical dynamics, which avoids the repetition (and concomitant errors) of the equations that each variant has in common. For example, we extend `baseMachine` in the `Order2` of Listing 3 in Line 2 to add Equation (6), as shown in Lines 10-11. The base model is also modified with start values e'_q (named `vf`, see Line 2), as well as computed parameters in Lines 4-8 that simplify the symbolic implementation of Equation (6).

E. System Model Composition

One of the key advantages of Modelica is that complex system-level models can be easily constructed by hierarchically connecting a set of component models. We demonstrate this process by building a system model that includes a generator unit, loads, and a power network to study the effect of faults on system dynamics. These models are available in the Github repository https://github.com/ALSETLab/CCTA-OpenIPSL, within the `Example1` package.

The network model is based upon a `partial model` in `Example1.Base.Networks.Base` that comprises buses `B1`-`B4`, transmission lines `line_1`-`line_3`, the transformer `transf` and the `System Data` component, see Fig. 3. As was the case in the machine model, different subsequent system model variants can be built by extending and modifing this partial base model. In this example, we first extend the base model twice by adding the remaining components in Figure 3 except for G1; this base model is named `Example1.Base.Networks.BasePFnFault`.

The generator unit model itself is constructed by connecting all the components required to model a specific power generator, i.e., the synchronous machine and its different regulators, as shown in Fig. 4. It is used in Fig. 3 as an instance called `G1`. It has three main components labeled red in Fig. 4, a power system stabilizer (PSS) that is a specialized controller that needs to be designed to help provide damping; an excitation control system (ECS) whose purpose is to regulate the terminal voltage of the generator; and finally,

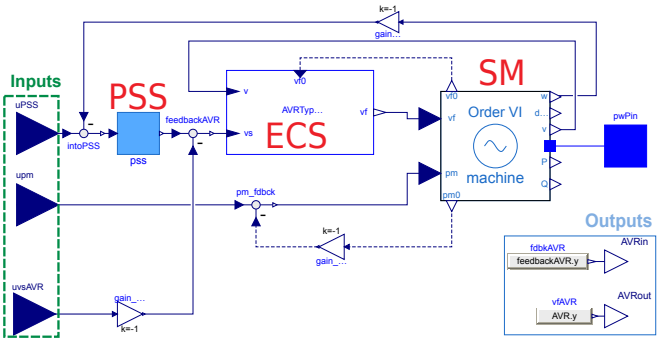

Fig. 4. `Example1.Base.Plants.GenIO` generator unit model with IO interfaces.

the synchronous generator (SM), which is a 4th-order model instantiated as `machine`.

The system model in Figure 3 can be constructed with the assembled network and generator unit models by extending the `BasePFnFault` and instantiating the `gridIO` model, as shown in Listing 4 (Lines 4-5). A set of inputs and outputs is defined to linearize the model with the desired input/output causality, e.g., defining the input for a probing signal in Fig. 3 `uvs` corresponding to `uvsAVR` in Fig. 4. In Line 3 an interface that defines outputs is instantiated and is linked to the output of each component in Lines 7-9. For example, the generator's speed `G1.machine.w` is linked to the interface `w` in Line 7; similar interface equations are also defined for the other variables of the generator unit.

Listing 4. Linking output variables to the `RealOutput` interfaces.

```
1  model gridIO
2    "Power grid model with input/output interfaces ..."
3    extends Example1.Interfaces.
         OutputsInterfaceWEfdAndAVRout;
4    extends Base.Networks.BasePFnFault(..., Plants.GenIO
         G1(P_0=pf.machines.PG1, ...);
5    ... (additional instantiations follow)
6  equation
7    w = G1.machine.w;
8    delta = G1.machine.delta;
9    AVRin = G1.feedbackAVR.y;
10   ... (additional connect statements follow)
11 end
```

III. Simulation and Linear Analysis

Simulation and linearization are among the most common uses for system models because of the opportunities they offer for analysis for system design and operation, for example, in control design [25], as existing domain-specific tools [26] generally cannot linearize power system models or require a deep understanding of the underlying source code to obtain useful results [27], the symbolic aspects of Modelica models provide significant advantages for these tasks. In particular, symbolic transformations can be used to readily transform the models between different sets of inputs and outputs, and numerical errors due to finite differences during the linearization of nonlinear models can be mitigated or eliminated. The symbolic underpinnings of the Modelica language also enables *automatic* derivation of linear models from *exactly* the same model used for nonlinear time-domain simulation to extract arbitrary state-space representations of power system models at any operating condition and point in time, which is not possible with domain-specific tools.

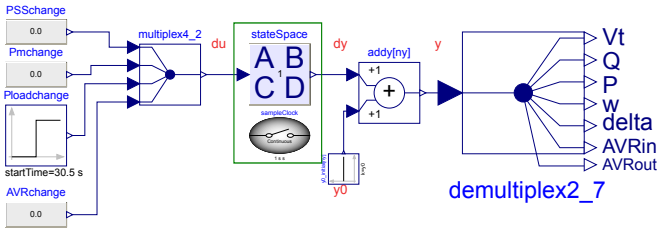

Fig. 5. `Example1.Analysis.LinearAnalysis.LinearModel General` configured for simulation of a step response on `Ploadchange`.

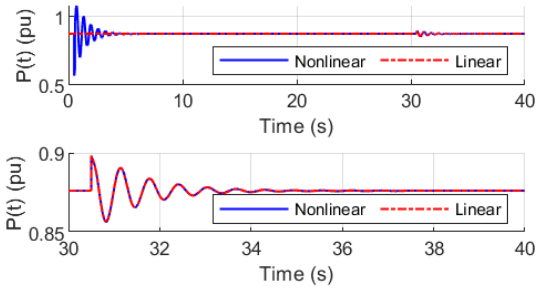

Fig. 6. Comparison of model responses under a load disturbance at at t=30.5 sec. Top: 0-40 sec., Bottom: 30-40 sec.The linear model is obtained by simulating the nonlinear-model until t=30.5 sec and linearizing it at that time.

Moreover, the behavior of the resulting linear models can be directly compared to that of the nonlinear models to assess the quality of the linearization, as models are available in the MSL that allow state-space models to be automatically loaded and simulated for such comparisons. As an example of this workflow, we configure the models in `Example1.Analysis.LinearAnalysis` to simulate two changes: 1) a large disturbance that removes `line_4` at t=0.5 sec and 2) a large load increase, modeled with a `pulse` at t=30.5 sec. The first change effectively modifies the linearized system state space matrices of the system obtained from the original model including `line_4`, necessitating linearization both before and after the line is removed. Comparison of the outputs of the linear and nonlinear models during the transient load after the second change allows the quality of the linear model to be assessed against the non-linear models. We illustrate the output of the linear model and the nonlinear model at t = 30.5 s in Fig.6 to demonstrate the ability to easily carry out such studies and use the linearized model as the ground truth in Section IV.

The Modelica language and its associated tools also provide a set of specific attributes designed to improve the numerical performance of nonlinear simulations. As consistent start values are required to solve a set of DAEs, the `start` variable attribute allows the model creator to provide important information that can be used to initialize the models, from which a Modelica tool can calculate consistent values for all other system variables. OpenIPSL has been specifically designed to allow the user to provide readily-available data to calculate the starting values inside each of the components by using a Modelica `record` template that is mapped to each of the components of the model, so that start values can be automatically calculated from the power flow solution data [28].

The solvers provided in Modelica tools also provide important flexibility and advanced capabilities to simulate large and numerically stiff systems of equations that represent power systems. While most domain-specific tools are limited to specific solvers that use fixed time steps and, as such, are practically limited to simulations of a few minutes, the variable-step solvers in Modelica tools can use modern variable-step ODE/DAE solvers [29] and take advantage of sparsity to bypass such restrictions. Together with advances in multi-core parallelization [30], this has led to major improvements in simulation performance of Modelica tools; for power systems, [29] shows that Modelica tools are competitive, even outperforming power system simulators in certain scenarios. These capabilities allow to perform simulations with exogenous time-series inputs from experimental data and stochastic processes, with reproducible models of noise with pseudo-random numbers for uniform, normal, truncated, and band-limited noise. The use of exogenous inputs is valuable for system identification purposes [15], while stochastic modeling for power systems has proven to be useful for the design of control and protection schemes for the resynchronization of islanded grids, as shown in [31]. Such capabilities are not available in conventional power system tools.

IV. Identification and Control Use Cases

A. Preliminaries

Identification and control methods are instrumental in addressing the dynamic performance issues faced by power systems. We illustrate how Modelica and the OpenIPSL can be used to design methods targeting such issues by focusing on local oscillation monitoring and damping enhancement [25] for power plants.

One countermeasure to deal with such oscillations and "wide-area" behavior such as inter-area modes [32] equips different power plants with a so-called power system stabilizer (PSS). PSSs are placed before the voltage control loop (i.e., the excitation control system (ECS)) in synchronous machines. This allows the ECS and the generator to be exploited to increase the damping from the plant and ensure the stability of the connection to the grid. However, these damping controllers must be well tuned to maintain adequate damping and may lead to major grid stability problems without calibration [33]. The PSS structure (a transfer function) is fixed and the value of its parameters must be set through design studies before being implemented in the field. The need to maintain validated models represents a major challenge, as seen recently in the European interconnected system [33]. We thus consider the power system as a *digital twin* of the grid where experiments can be conducted.

In practice, this approach is attractive because the increase in the availability of power system measurement devices, e.g., phasor measurement units (PMU), allows oscillation monitoring [34]. Although "wide area" dynamics have been the main focus, it can also be beneficial to monitor and potentially minimize local dynamics such as those recently observed in [35]. In [15]) a data-based PSS redesign method was recently proposed. In this work, we extend the approach

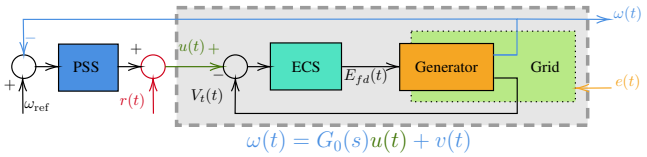

Fig. 7. Block Diagram of a Power Plant's Electrical Generator Equipped with a PSS Interconnected with a Power Grid.

to consider multiple operating points (MOPs) under "ambient conditions". Such methods can be very attractive to plant owners and operators in monitoring and maintaining satisfactory damping performance levels at all times and under all operating conditions.

B. Problem Statement

The identification and control problem to be solved is specified by using the diagram in Fig. 7. This illustrates a power plant similar to that modeled in Fig. 4, which is connected to a power grid. In Fig. 7, the PSS derives a damping signal $u(t)$, which is applied to the ECS that modulates E_{fd}, the machine field winding voltage, to damp oscillations while also controlling the terminal voltage magnitude $V_t(t)$. Observe that the impact of any changes in the power grid will be reflected in $V_t(t)$ and $\omega(t)$. Meanwhile, the signal $v(t)$ represents the influence of random load changes $e(t)$ on $\omega(t)$. Finally, the signal $r(t)$ will be applied to identify a data-based model of the power system defined by

$$\omega(t) = G_0(s)u(t) + v(t)\,,\; u(t) = -K(s)\omega(t) + r(t) \quad (7)$$

where $K(s)$ is the continuous-time transfer function (TF) of the PSS controller, and with $\omega(t)$.

In (7), $G_0(s)$ represents the dynamics of the power system between $u(t)$ and $\omega(t)$ and therefore embeds the dynamics of the ECS, generator and the grid. At each operating point, it is assumed that these dynamics can be represented by a linear TF $G_0(s)$. Note that the disturbance of the process $v(t)$ represents the effect of random load changes on $\omega(t)$ and is considered filtered white noise. Using (7), the following expression of $\omega(t)$ in closed loop is:

$$\omega(t) = T_0(s)r(t) + M_0(s)v(t), \quad (8)$$

where

$$T_0(s) = G_0(s) \;/\; (1 + K(s)G_0(s)), \quad (9)$$

$$M_0(s) = 1 \;/\; (1 + K(s)G_0(s)). \quad (10)$$

These definitions allow us to summarize the following three sub-problems (see details of each of these in [15]):

P1. Oscillation monitoring aims to evaluate the damping ability of the PSS controller, i.e., the plants ξ_{min} for the local mode. This requires verifying that all the complex poles of the closed-loop system (8) have a damping coefficient greater than a given threshold ξ_{req}.

P2. If the estimated ξ_{min} (and its uncertainty interval) are deemed not satisfactory (smaller than ξ_{req}), a model of G_0 is obtained using a probing signal $r(t)$ to redesign the PSS. This gives the model $\hat{T}$ of T_0, which is used to calculate $\xi_{min}(\hat{T})$ and to verify whether a controller update is really necessary.

P3. If the damping is deemed insufficient. The PSS controller is redesigned using a model $\hat{G}$ of G_0.

Observe that for redesign using (9), this model can be deduced from $\hat{T}$ via:

$$\hat{G}(z) = \frac{\hat{T}(z)}{1 - K(z)\hat{T}(z)} \quad (11)$$

where $K(z)$ is the discrete-time version of the current PSS controller $K(s)$. This gives $\hat{G}$, a model of the open-loop system G_0.

C. Modeling, Simulation and Linearization Scenarios

The system model in Fig. 3 is an updated and improved version of the model used in [15], where the P1-P3 are solved in the case where the grid undergoes a disturbance (i.e. loss of `line_4`). Expanding on that previous work, in this paper, we extend the proposed approach to consider MOPs under ambient conditions for a more complex power system shown in Fig. 8 that was first proposed in [32].

The model in Fig. 8 was developed to solve the identification and control problems defined above and is provided within the `Example2` package. Due to space constraints, a detailed modeling description is omitted here; however, it is worth noting that it follows the approach used in developing `Example1` in Fig. 3.

To simulate MOPs, i.e., different dispatch points at which the plant is operated, the required power output of the plant is varied by ramping $u_{P_m}(t)$ (see the red dashed square in Fig. 8), while at the same time increasing the power demand through $e_{Load9}(t)$. The simulation scenarios shown in Fig. 9 are designed so that as the power dispatch (and load) is increased, the system's damping will reduce, and vice versa, while at the same time exciting the system's dynamics through random load changes.

The power plant operates at $P \approx 7.00$ per unit (p.u., 100 MVA base) at $t_A = 0$ min. and transitions to a new OP at $t_\alpha = 7.5$ min. As the system moves to different OPs, the power system approaches equilibrium at $t_{\alpha,\beta,\gamma,\delta,\psi} = [7.5, 25.0, 40.0, 55.0, 67.5]$ where the power dispatch is $P_{\alpha,\beta,\gamma,\delta,\psi} = [7.83, 8.29, 8.02, 7.73]$. The model is linearized at each of these operating points using the features described in Section III and the data collected from the simulations at each of these equilibria is used to determine the damping according to P1 in Section IV-B. Meanwhile, to solve P2, $r(t)$ is applied in the periods $t_{B-C} = [12.5, 17.5]$, $t_{D-E} = [27.5, 32.5]$, $t_{F-G} = [42.5, 47.5]$ and $t_{H-I} = [57.5, 62.5]$. Finally, the PSS redesign that is carried out to solve P3, will take place in t_{K_x}, if necessary, where $x = 0, 1, ..., 4$ and K_0 is the original PSS design.

D. System Identification for Oscillation Monitoring

Oscillation monitoring (P1) requires the determination of a time series model of $M_0(s)v(t)$ in (8) using discrete-time data $\omega[n]$ collected during "ambient" OPs. The discrete-time sequence $\omega[n]$ can indeed be modeled as $\omega[n] = H_0(z)e[n]$

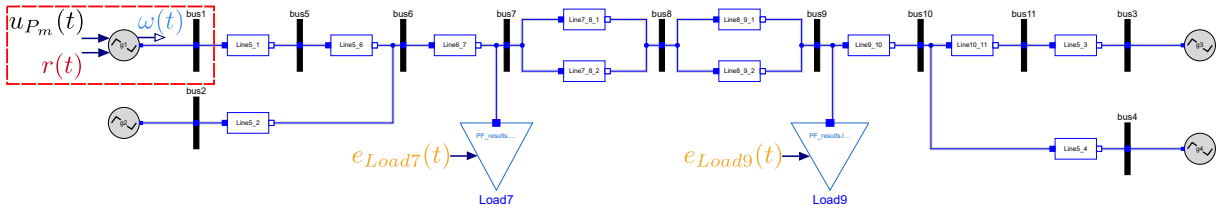

Fig. 8. Two-Area Four-Machine Klein-Rogers-Kundur Power System Model.

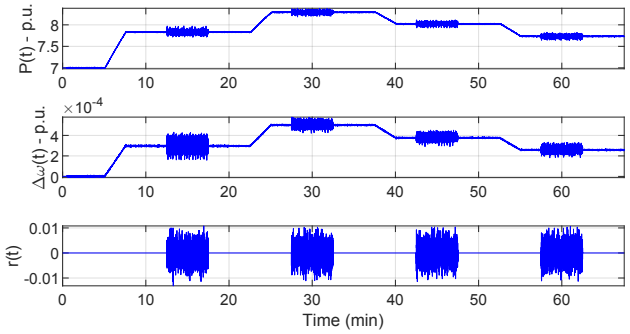

Fig. 9. Simulation of the Model in Fig. 8 under Ambient Conditions and Multiple Operating Points

where $H_0(z)$ is a discrete-time monic, stable, inversely stable transfer function and $e[n]$ a white noise.

Using the discrete-time data $\omega[n]$, we can identify an ARMA model $\hat{H}(z)$ of H_0 (and an estimate of the variance of $e[n]$). Thus, the performance of the existing PSS design K can be evaluated. Next, we inspect the damping of the complex poles of $\hat{H}$ and determine the minimal value of their damping, i.e., $\xi_{min}(\hat{H})$ which is an estimate of ξ_{min} of the loop (7). Furthermore, we can also determine an uncertainty interval around this estimate (see [34] for details). Using the data collected from $\omega[n]$ between t =[7.5,12.5] min., $\hat{H}$ was estimated as shown in Fig. 10, where it is compared to the true H_0 (red) obtained from the linearized model. $\hat{H}$ was obtained using an ARMA model structure from MATLAB's System Identification Toolbox by setting the `armax` function with `na=nc=8`. $\hat{H}$ gives a fit of 87.74%, which is excellent considering the low model order of 8 used compared to the higher order of H_0 which contains 52 states. More importantly, $\hat{H}$ contains the dominant oscillation's frequency, 7.7 rad/sec (≈1.22 Hz), which exists in H_0. Using a higher-order model does not give better results, as there are several modes that are not observable by $\omega(t)$ in any case (see [36]). The damping and its confidence interval are:

$$\xi_{min}(\hat{H}) = 0.0564 \text{ and } \mathcal{I}_{\hat{H}} = \{0.0,\ 0.0876\}.$$

The true value is ξ_{true}=0.0591 and what it is obtained from Equation $\xi_{min}(\hat{H})$ is ≈ 0.5 % lower, while the true value is contained within the upper bound of $\mathcal{I}_{\hat{H}}$. This indicates that the damping is insufficient for a desired ξ_{req}=15%.

Having found that the damping is not adequate, we proceed to solve P2. The probing signal $r(t)$ (a multisine) is applied. Data for $\omega[n]$ and $r[n]$ from t =[12.5,17.5] min are collected to estimate $\hat{T}$ using a Box-Jenkins model with parameters `nb=nc=nd=nf=6` and `nk=0`, achieving a fit of 89.84%. The estimated model $\hat{G}$ obtained from (11) using $\hat{T}$ and K_0 and is shown in Fig. 11 compared to the true

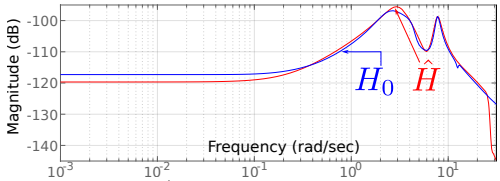

Fig. 10. H_0 and $\hat{H}$ estimated from $\omega[n]$ t=[7.5,12.5] min.

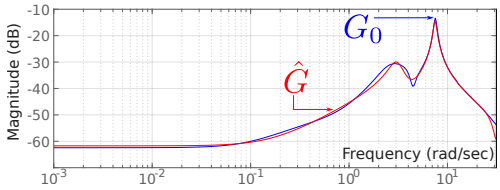

Fig. 11. G_0 and $\hat{G}$ estimated from $\hat{T}$ and K_0 for t=[12.5,17.5] min

G_0, where it can be seen that $\hat{G}$ matches G_0 (especially at 7.7 rad/sec).

E. Control Re-Design for Oscillation Damping

Using $\hat{G}$ identified above (see Fig. 11), we proceed to solve P3, which provides a new PSS design. Applying the method proposed in [15] gives a new controller K_1 with parameters $k_{w,1}$=49.6273 and $t_{w,1}$=0.3801 sec. Before applying this design, the new damping that could be achieved with it computed from

$$\xi_{min}(K_1, \hat{G}) = 0.3230 \ \& \ \mathcal{I}_{new,1} = \{0.2741, 0.3305\},$$

which are larger than ξ_{req}=15.00%. We consequently decided to apply the new PSS design at t=20 min., as it provides a damping of 32.30% with bounds larger than ξ_{req}.

The original PSS parameters, K_0, are now replaced by the new controller K_1. While the controller operates under ambient conditions, it is possible to use ambient data to verify the performance of K_1, that is, to verify that $\xi_{min}(K_1, G_0)$ is satisfactory. This can be achieved by solving P1 again using data from t=[1200-1350]. We thus use an ARMA model with `na=nc=8` to estimate $\hat{H}_{K1}$ using blind identification which gives $\hat{H}_{K1}$ shown in Fig. 12. In Fig. 12, $\hat{H}_{K1}$ is compared with H_{0,K_1}, $\hat{H}_{0,K_0}$ and H_{0,K_0} (which are also shown in Fig. 10). Comparing $\hat{H}_{0,K_0}$ with $\hat{H}_{K_1}$ reveals that the designed controller effectively removed the peak at 7.7 rad/sec (≈1.22 Hz) by providing substantial damping.

As can be observed in Fig. 12, the mode in $\omega = 7.7\ rad/s$ (which is the result of interactions between the power plants `g1` and `g2`) is no longer the one with the smallest damping. The mode with the smallest damping is a lower frequency mode at ≈2.4 rad/sec. This is the so-called inter-area mode (see [32]). The model $\hat{H}_{K_1}$ can be used to see that this mode has a damping equal to 24.74%, i.e., $\xi_{min}(\hat{H}_{K_1}) = 24.74\%$.

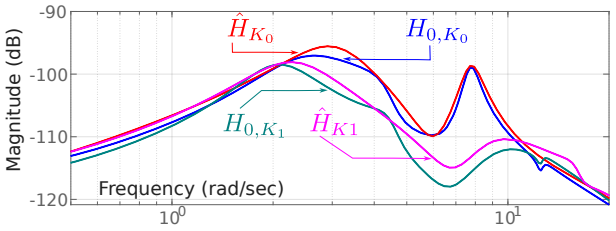

Fig. 12. Assessment of the application of $K_{new} = K_1$

This confirms that K_1 has a sufficient damping capacity at the new operating point. Further analysis of the performance of the redesigned controller at the other operating points is omitted here and is provided in [16].

V. Conclusion

This paper has illustrated how power system models can be developed using Modelica and OpenIPSL in system identification and control applications. In the particular identification and control methods presented here, the models were used as a *digital twin* of a power system in which probing experiments were performed to perform oscillation monitoring and damping control redesign. This is useful because there are limited opportunities to perform probing experiments in actual power grids and, more importantly, where the "ground truth" (e.g., the precise damping values) is unknown. This in turn helps the development process of data-driven applications, making them ready for later stages, such as in *ex situ* testing in laboratory facilities where Modelica models can be reused [37].

Acknowledgement

The authors thank the contributors to OpenIPSL and the support of the funding bodies that have facilitated its development, which are listed in http://openipsl.org, and without whom the library would not exist. The contributions of X. Bombois in the development of the methods described in [15] are duly recognized.

References

[1] A. Isaacs, "Simulation technology: The evolution of the power system network," IEEE Pwr. Energy Mag., vol. 15, no. 4, pp. 88–102, 2017.
[2] K. Prabhashankar and W. Janischewsyj, "Digital simulation of multi-machine power systems for stability studies," IEEE Trans. Power App. Syst., vol. PAS-87, no. 1, pp. 73–80, Jan. 1968.
[3] H. W. Dommel, "Digital computer solution of electromagnetic transients in single-and multiphase networks," IEEE Trans. Power App. Syst., vol. PAS-88, no. 4, pp. 388–399, Dec. 1969
[4] "PSS®E 34.2.0 Model Library," Siemens PTI, Schenectady, NY, 2017.
[5] J. Mahseredjian *et al.*,, "Electromagnetic Transients Simulation Program: A unified simulation environment for power system engineers," in IEEE Electrification Mag, vol. 11, no. 4, pp. 69-78, Dec. 2023.
[6] F. Fachini *et al.*, "Modeling and Validation of Renewable Energy Sources in the OpenIPSL Modelica Library," in 47th Annual Conf. of the IEEE Industrial Electronics Soc., Toronto, ON, Canada, 2021, pp. 1-6.
[7] J.D. Lara *et al.*, "PowerSystems.jl — A power system data management package for large scale modeling," SoftwareX, Volume 15, 2021, 100747, ISSN 2352-7110.
[8] L. Vanfretti *et al.*, "Unambiguous power system dynamic modeling and simulation using modelica tools," 2013 IEEE PES GM, Vancouver, BC, Canada, 2013, pp. 1-5, doi: 10.1109/PESMG.2013.6672476.
[9] B. Bachmann and H. Wiesmann, "Advanced Modeling of Electromagnetic Transients in Power Systems," in Modelica Workshop 2000 Proc., Oct. 23, 2000, pp. 93–97.
[10] D. Winkler, "Electrical Power System Modelling in Modelica - Comparing Open-source Library Options", in Proceedings of the 58th Conf. on Simulation and Modelling, 2017, vol. 138, pp. 263–270.
[11] A. Guironnet *et al.*, "Towards an Open-Source Solution using Modelica for Time-Domain Simulation of Power Systems", in 2018 IEEE PES ISGT Conf. Europe, 2018.
[12] A. Masoom *et al.*, "MSEMT: An Advanced Modelica Library for Power System Electromagnetic Transient Studies," in IEEE Trans. Power Deliv., vol. 37, no. 4, pp. 2453-2463, Aug. 2022.
[13] A. Senkel *et al.*, "Status of the TransiEnt Library: Transient Simulation of Complex Integrated Energy Systems," in Proc. of 14th Modelica Conf. 2021, Linköping, Sweden, September 20-24, 2021. DOI: 10.3384/ecp21181187
[14] Marcelo de Castro *et al.*, "Version [OpenIPSL 2.0.0] - [iTesla Power Systems Library (iPSL): A Modelica library for phasor time-domain simulations]", SoftwareX, Volume 21, 2023, ISSN 2352-7110.
[15] X. Bombois and L. Vanfretti, "Performance monitoring and redesign of power system stabilizers based on system identification techniques," Sustainable Energy, Grids and Networks, Volume 38, 2024, 101278, ISSN 2352-4677.
[16] L. Vanfretti and X. Bombois, "Power System Oscillation Monitoring and Damping Control Re-Design under Ambient Conditions and Multiple Operating Points," in Proc. 20th IFAC Symposium on System Identification, Boston, MA, USA, July 17-19, 2024.
[17] M. de Castro and L. Vanfretti, "Multi Time-Scale Modeling of a STATCOM and Power Grid for Stability Studies using Modelica," 2022 OSMSES, Aachen, Germany, 2022, pp. 1-7.
[18] M. Aguilera *et al.*, "Coalesced Gas Turbine and Power System Modeling and Simulation using Modelica," American Modelica Conf. 2018, pp. 93-102, October 9-10, Cambridge, MA. doi: 10.3384/ecp1815493
[19] Modelica Association, "Modelica Standard Library v4.0.0, 2020-06-04, Available online: http://tinyurl.com/mslv4 , Accessed: Feb. 18, 2024.
[20] Modelica Association, "Modelica Language Spec. v3.6.0, 2023-03-09, Available online: http://tinyurl.com/mspc36 , Accessed: 18. Feb. 2024.
[21] M. de Castro *et al*, "A Fundamental Time-Domain and Linearized Eigenvalue Analysis of Coalesced Power Transmission and Unbalanced Distribution Grids using Modelica and the OpenIPSL," in Proc. 13th Intl. Modelica Conf., Regensburg, Germany, 2019. doi: 10.3384/ecp19157617.
[22] G. Laera *et al.*, "Guidelines and Use Cases for Power System Dynamics Modeling and Model Verification using Modelica," American Modelica Conf. 2022, October 26-28, Dallas, Texas. doi: 10.3384/ECP21186146.
[23] F. Milano, *Power System Modeling and Scripting*. Springer-Verlag, Berlin, Heidelberg, 2010.
[24] F. Milano, "An open source power system analysis toolbox," in IEEE Trans. on Power Systems, vol. 20, no. 3, pp. 1199-1206, Aug. 2005, doi: 10.1109/TPWRS.2005.851911.
[25] G. Rogers, "The effect of power system stabilizers on system performance," in Proc. 1999 IEEE Power Eng. Soc. Summer Meeting (Cat. No.99CH36364), Edmonton, AB, Canada, 1999, pp. 110-115 vol.1.
[26] N. Nikolaev *et al.*, "PSS/E Based Power System Stabilizer Tuning Tool," 2020 21st Intl. Symposium on Electrical Apparatus & Tech., Bourgas, Bulgaria, 2020, pp. 1-6, doi: 10.1109/SIELA49118.2020.9167137.
[27] W. Li *et al.*, "Development and Implementation of Hydro Turbine and Governor Models in a Free and Open Source Software Package," Simulation Modelling Practice and Theory, vol. 24, pp. 84-102, 2012.
[28] S.A. Dorado-Rojas *et al.*, "Power Flow Record Structures to Initialize OpenIPSL Phasor Time-Domain Simulations with Python," in Proceedings of the 14th Intl. Modelica Conf., 2021, pp. 147–154.
[29] Erik Henningsson *et al.*, "DAE Solvers for Large-Scale Hybrid Models", Proceedings of the 13th Intl. Modelica Conf., Regensburg, Germany, March 4–6, 2019, ISSN 1650-3740.
[30] H. Elmqvist *et al*, "Parallel Model Execution on Many Cores," in Proc. 10th Intl. Modelica Conf., March 10-12, 2014, Lund, Sweden, doi: 10.3384/ecp14096363
[31] B. Mukherjee *et al.*, "A PMU-Based Control Scheme for Islanded Operation and Re-synchronization of DER," Intl. Journal of Electrical Power Energy Systems, Volume 133, 2021, 107217, ISSN 0142-0615.
[32] M. Klein *et al*, "A fundamental study of inter-area oscillations in power systems," in IEEE Trans. on Power Syst., vol. 6, no. 3, pp. 914-921, Aug. 1991.
[33] ENTSO-E Sub-Group System Dynamics and Protection, "Analysis of CE Inter-Area Oscillations of 1st December 2016," *ENTSO-E*, 2017, Available online: https://tinyurl.com/ENTSOE2016
[34] V. S. Perić *et al*, "Optimal Signal Selection for Power System Ambient Mode Estimation Using a Prediction Error Criterion," in IEEE Trans. Power Syst., vol. 31, no. 4, pp. 2621-2633, 2016.
[35] C. Mishra *et al.*, "Analysis of STATCOM Oscillations using Ambient Synchrophasor Data in Dominion Energy," 2022 IEEE PES ISGT Conf., New Orleans, LA, USA, 2022, pp. 1-5.
[36] L. Vanfretti and J. H. Chow, "Analysis of power system oscillations for developing synchrophasor data applications," 2010 IREP Symposium Bulk Power System Dynamics and Control - VIII, Rio de Janeiro, Brazil, 2010, pp. 1-17, doi: 10.1109/IREP.2010.5563289.
[37] M. de Castro *et al.*, "Real-Time Prototyping of Optimal Experiment Design in Power Systems using Modelica and FMI," 2022 IEEE PES GM, Denver, CO, USA, 2022, pp. 1-5, doi: 10.1109/PESGM48719.2022.9916801.