Introduction to OmniPES: a Novel Modelica Library for Power Systems Modeling and Analysis

Marcelo A. Tomim, Ricardo M. Henriques, João A. Passos Filho
Department of Electrical Energy, Federal University of Juiz de Fora, UFJF, Juiz de Fora, Brazil
marcelo.tomim@ufjf.br, ricardo.henriques@ufjf.br, joao.passos@ufjf.br

***Abstract*—This paper introduces OmniPES, a Modelica library tailored for the swift prototyping of small to medium-scale power systems, facilitating analysis in both steady-state and transient conditions. Notable features include sigmoid-based switches that enforce reactive power constraints in power sources during steady-state or quasi-steady-state operation. Moreover, the library incorporates a transient stability modeling framework, offering a structure akin to those utilized in production-grade power systems programs. These features are illustrated through a tutorial system operating under a secondary voltage regulation loop. The presented results validate and clearly demonstrate the accuracy and effectiveness of the proposed library.**

***Index Terms*—power systems, modelica, secondary voltage control, steady-state analysis, transient analysis.**

Acronyms

AVR	Automatic Voltage Regulator
CVRMSE	Coefficient of the Variation of the Root Mean Square Error
DER	Distributed Energy Resources
IEA	International Energy Agency
LTC	Load Tap Changer
MRE	Maximum Relative Error
MSL	Modelica Standard Library
OECD	Organization for Economic Co-operation and Development
PI	Proportional-Integral
PSS	Power System Stabilizer
SR	Speed Regulator
SVR	Secondary Voltage Regulator

I. Introduction

Electric power system operators have been facing ever-increasing challenges due to the rapid growth in the utilization of distributed energy resources, often associated with renewable energy sources. According to the International Energy Agency (IEA), electricity generated by renewables in the Organization for Economic Co-operation and Development (OECD) countries [1] increased by 80.8 % from 2010 to 2022 at a rate of approximately (128.0 ± 2.6) TWh/year. In contrast, non-renewables generated electricity experienced a 13.8 % decrease in the same period averaging around (97.7 ± 9.2) TWh/year [2]. On a down side, however, renewable energy resources are naturally dispersed throughout electric power grids and often suffer from intermittency, imposing additional challenges for system operators to predict their energy availability and integrate them seamlessly with dispatchable energy resources, such as non-renewables and/or hydro power with reservoirs.

Distributed energy resources (DER) are often connected to electric networks through power converters, granting DERs the capability to contribute to the voltage and frequency regulation of the system. These devices, however, require more advanced modeling and analysis techniques, which must be incorporated into more traditional analytical frameworks, such as transient stability analysis.

In such circumstances, engineering students and professionals can enhance their understanding of power system dynamics by using tools that enable them to rapidly prototype models and integrate them into larger systems, supplementing more traditional approaches. As an example of such tools, the Modelica language was developed with the intent to standardize the mathematical description of complex dynamic systems, which combines continuous and discrete domains [3].

Several Modelica libraries that provide frameworks for power system analysis are available [4], [5], [6], [7], not to mention the variety of models provided by the Modelica Standard Library (MSL) [8]. In the present development, one of the first intentions was to learn how the Modelica language could be effectively used for power system elements modeling and analysis. As the library matured, it was structured along the same lines as industrial-grade power-flow and transient stability programs employ for modeling bulk power systems. At the present stage of the development, the OmniPES library provides mainly two analysis frameworks: steady-state and transient stability. In the first framework, power-flow restrictions are enforced at all times during a simulation. The second one, on the other hand, considers the premises of transient stability programs, which allow the inclusion of slow dynamics associated with generation, load, and other system controlling devices.

Furthermore, within the OmniPES library, initial operating conditions are established through embedded power-flow restrictions applied to power plants and loads. This modeling feature facilitates the rapid prototyping of small to medium-scale power systems, eliminating the necessity of importing power flow results from third-party programs.

Since the library is publicly available as a GitHub repository [9], in this article, the most fundamental features of the OmniPES library are presented with the aid of the tutorial system presented in [10] and also used by a CIGRÈ task force in [11]. This particular system was chosen so that the results obtained, both in dynamic and steady-state conditions, can be replicated and readily compared with those already published. Additionally, the Secondary Voltage Regulator (SVR) implemented for this system has allowed us to showcase the level of flexibility achievable with the proposed Modelica library for modeling both conventional and unconventional power systems.

The main contributions of this work are twofold. Firstly, the generators' reactive power limits are modelled using sigmoid-based switches, as described in [12], within the Modelica language. This model enables precise restriction of reactive power output by the sources while adjusting the controlled variables as required by the system's solution, including the backoff procedure from reactive power limits. Secondly, initial conditions for transient or quasi-steady-state simulations are implicitly defined by embedded power-flow restrictions, eliminating the need for users to import power flow results from external third-party tools.

In the paper's sequence, steady-state simulations for the tutorial system are first discussed, examining its operation with and without secondary voltage regulation when subjected to a load increase. Subsequently, dynamic simulations for the same tutorial system are presented. In addition to the preceding discussion on simulation preparation, it is also explored the object-oriented strategy employed for modeling power plants, which can serve as a template for modeling other devices in the future.

II. Package Description

In Fig. 1(a), an overview of the OmniPES library is shown. Among its subpackages, one can identify the subpackages `Circuit`, `SteadyState`, and `Transient` as the most important ones in the library, while the others contain auxiliary models and tools such as mathematical functions, unit definitions, and scopes.

The `Circuit` subpackage contains the most basic network elements for positive-sequence power system analysis, such as electrical buses, shunt and series components used to construct more complex models such as voltage and current sources, transmission lines, two-winding transformers, breakers, and faults.

In the `SteadyState` subpackage, one can find models that, together with the `Circuit` subpackage, allow one to perform traditional power flow analysis based on positive-sequence equipment models. In this package, the focus remain on loads and sources, typically specified by means of active and reactive power and voltage magnitudes. Models for various types of power sources are available such as Vθ, PV, PQ, and PQ with reactive power limits, in addition to non-linear polynomial load, the well-known ZIP loads. In order to avoid any misinterpretation, the models available in the `SteadyState` subpackage allow power flow computations at every time instant. In this context, the `time` can be seen as a parameterization of model variations, which can account for daily load profiles, generation ramps, wind speed profiles, etc.

The `Transient` subpackage provides models for electromechanical stability studies, such as stability models for synchronous machines and their associated controllers, such as voltage and speed regulators, in addition to power system stabilizers and polynomial loads. One important aspect that cannot be disregarded when dealing with electromechanical stability simulations is the establishment of proper steady-state operating conditions. In the OmniPES library, the choice was to embed the power flow restrictions alongside the initial conditions equations, which distinguishes the present development from other power system libraries. In this manner, the user is not required to import initial operating conditions calculated by third-party programs.

Moreover, by appropriately combining models from `SteadyState` and `Transient` subpackages, the OmniPES library also makes it possible to prepare quasi-steady-state simulations to help analyze system behavior in the long-term time range [13]. Such simulation kind usually neglect faster dynamics (e.g., generator damper windings and inner current control loops in power electronic devices) and retain slower dynamics, such as those related to secondary voltage and/or frequency regulation, boiler control, under-load tap changers, minimum and maximum excitation limiters, load dynamics, among others.

Up to this moment, the OmniPES library has been fully developed with the OpenModelica modeling and simulation environment [14]. The latest simulations were all performed using the version `1.22.2 12-g3b7ae01`. The modelling implementation is based on the recent MSL 4.0.0 [8].

In the next sections, a test system is used to showcase the most fundamental features of the OmniPES library and its potential for the development of more complex systems such as those penetrated by power electronics-based equipment.

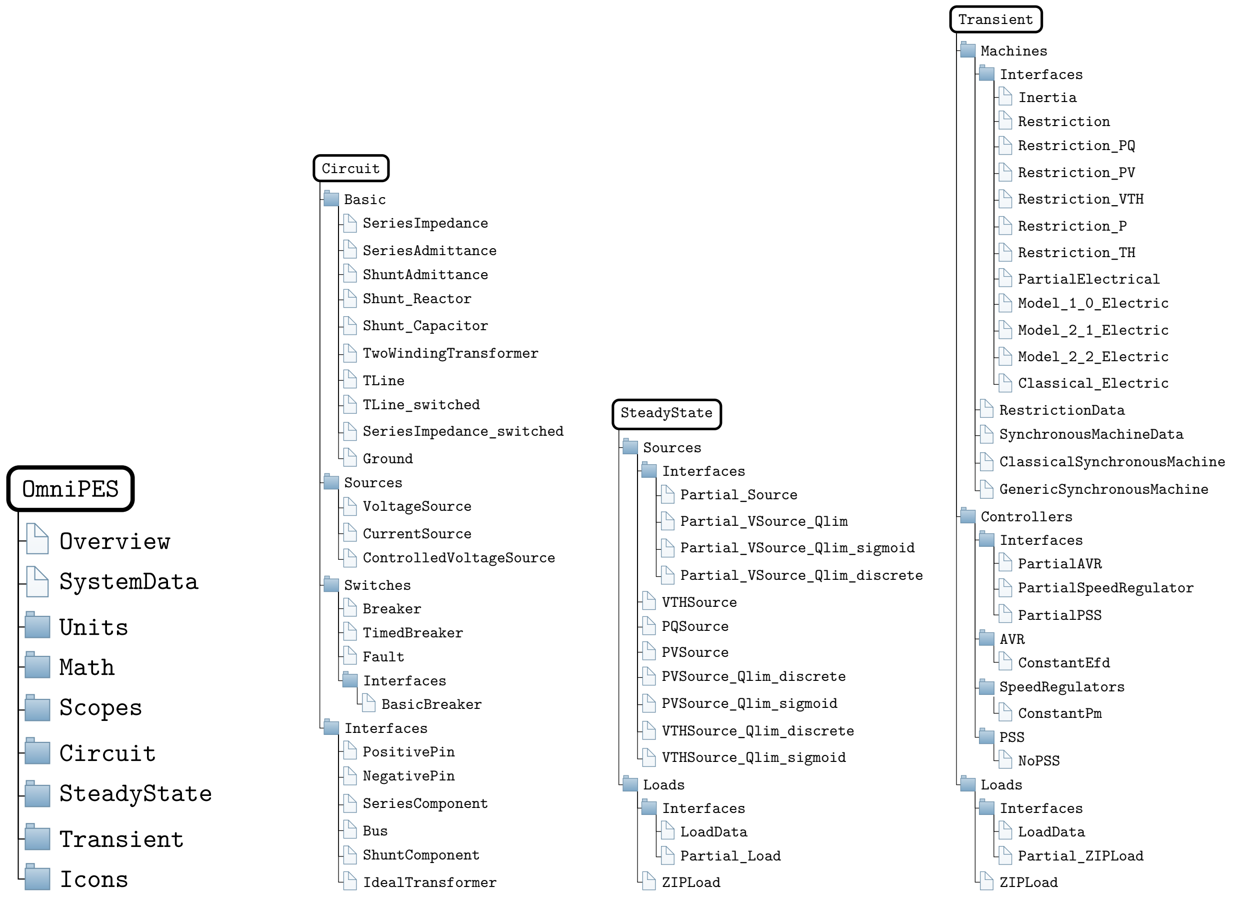
III. Tutorial System

As described in [10], [11], the present tutorial system consists of two power plants that supply a load through a transmission system formed by five branches (three transmission lines and two transformers) and five electrical nodes as shown in Fig. 2. This single-phase diagram was produced in the graphical interface OMEdit, which is part of the OpenModelica simulation platform [14].

The employed SVR scheme consists of a central controller and distributed plant controllers, all based on PI-controllers, which aims at maintaining a specific system bus voltage around a certain magnitude while keeping the reactive power share between the plants proportional to their nominal capacity.

IV. Steady-state Analysis

The steady-state analysis is the most commonly used tool in various studies conducted in electrical power systems. It primarily involves the calculation of nodal voltages, and the power flows in the transmission system, given a specified load level, established active generation dispatch, as well as the network's topology and parameters. In these conditions,



(a) OmniPES Overview (b) Circuit subpackage (c) Steady-state subpackage (d) Transient subpackage

Fig. 1: OmniPES library structure.

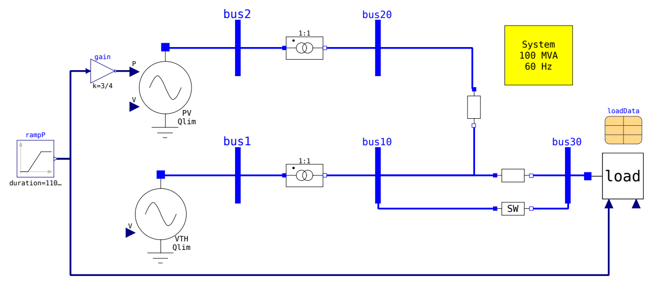


Fig. 2: Tutorial system implemented for steady-state analysis.

both active and reactive powers injected in every node can be calculated by means of the system nodal admittance matrix.

The solution of all complex voltages in a system is usually performed by "power-flow" or "load-flow" programs which model, in a systematic manner, active and reactive powers injected in each electrical node in terms of nodal complex voltages. Most of the current general-purpose power flow analysis programs use various variations of the Newton-Raphson method for its solution [15].

In the traditional power flow formulation [15], each electrical node contributes to the steady-state operating condition definition with two power equations (active and reactive) and four unknowns, i.e., real and imaginary components of both complex injected power and nodal voltage. So, for a system with n electrical buses, $2\,n$ equations need to be solved for $2\,n$ unknowns, whereas $2\,n$ variables need to be specified. Every electrical island, however, requires, at least, an angle reference, because active and reactive powers flowing through electrical branches (e.g., transmission lines and transformers) depend only on angle differences between adjacent nodes. Furthermore, the reference bus is also necessary for balancing the power mismatches in the system, including the system losses that are dependent on the currents flowing through the system. This is accomplished by adopting a reference bus for the system, also known as a "swing" or "slack" bus.

So, for each power system, one has to define the slack node, where nodal voltage magnitude and angle are known. For the rest of the electrical nodes, two out of the four variables (voltage magnitude and angle, and injected active and reactive powers) need proper definition. For example, traditional PQ nodes are those where injected active and reactive powers are known, whereas the complex voltage is fully unknown.

With this aspect in mind, firstly, `PartialSource` model was implemented as shown in Code 1(a), with two power equations (active and reactive power in line 17) and four unknowns (real and imaginary components of both complex injected power and nodal voltage, defined in lines 9 and 10, respectively). Notice that `PartialSource` model itself is inherited from the `ShuntComponent` model (line 5 of Code 1(a)), that defines the equipment's drawn current and its voltage drop, detailed in Code 1(b). Therefore, in order to complete the `PartialSource` model, two more variables need specification, according to the users' needs. In the sequence, some of the power-flow-based sources defined in the OmniPES library and their specified variables are presented:

`VTHSource`: both voltage magnitude and angle are defined;

`PVSource`: active power and voltage magnitude defined, as shown in Code 1(c);

`PQSource`: both active and reactive powers defined, according to Code 1(d);

`PVSource_Qlim`: active power and voltage magnitude are defined, while reactive power falls inside its operating range. Otherwise, the reactive power equals the violated limit, while the voltage magnitude becomes an unknown. Further discussion on this source will be set forth in subsection IV-A.

As for the loads, the previously shown `PQSource`-type and the `ZIPLoad` models can be used. The latter define both active and reactive powers in terms of, at most, second-order polynomials of their terminal voltage magnitude, as shown in equations (1) and (2), where P_0 and Q_0 correspond to the active and reactive power demanded by the load at the voltage magnitude of V_0. The constants β_p, β_i, and β_z with $\beta \in \{p, q\}$ define the amounts of constant power, constant current, and constant impedance, respectively, composing both the active and reactive power demanded by the load. Additionally, to ensure that the total load remains unchanged, the sum of the constants associated with active or reactive power must equal unity. Finally, the terms ΔP and ΔQ represent variations that may be imposed on the active and reactive power demanded by the load, respectively, during a simulation.

$$P_{load} = (P_0 + \Delta P)\left[p_p + p_i\left(\frac{V}{V_0}\right) + p_z\left(\frac{V}{V_0}\right)^2\right] \quad (1)$$

$$Q_{load} = (Q_0 + \Delta Q)\left[q_p + q_i\left(\frac{V}{V_0}\right) + q_z\left(\frac{V}{V_0}\right)^2\right] \quad (2)$$

The above model is implemented as shown in Code 2(a), which is inherited from `Partial_ZIPLoad` model given in Code 2(b). One can notice that the `Partial_ZIPLoad` holds the most basic relationships including the complex power equation in line 18 and the computation of the voltage magnitude in line 19. Additionally, the total load can also be modified by the external signals `dpsp` and `dqsp`, if necessary, according to lines 20 to 29 of Code 2(b).

A. Reactive Power Limits

In the traditional formulation of the power flow problem, the equations related to reactive power are not included in the linear system solved at each iteration of Newton's method.

Code 1: Source model hierarchy.

(a) `Partial_Source` abstract model.

```
3  partial model Partial_Source
4    extends Icons.Vsource;
5    extends Circuit.Interfaces.ShuntComponent;
6    outer SystemData data;
7    import Modelica.ComplexMath.conj;
8    import Abs=Modelica.ComplexMath.abs;
9    Modelica.Units.SI.ComplexPerUnit S;
10   Real V(start=1, max=1.5, min=0.7);
     (auxiliary code)
16 equation
17   S = -v*conj(i);
18   V = Abs(v);
     (auxiliary code)
25 end Partial_Source;
```

(b) `ShuntComponent` abstract model.

```
3  partial model ShuntComponent
4    PositivePin p;
5    Modelica.Units.SI.ComplexPerUnit v(re(start=1));
6    Modelica.Units.SI.ComplexPerUnit i;
7  equation
8    v = p.v;
9    i = p.i;
10 end ShuntComponent;
```

(c) `PVSource` model.

```
3  model PVSource
4    extends Interfaces.Partial_Source;
5    import Abs=Modelica.ComplexMath.abs;
6    parameter Units.ActivePower Psp;
7    parameter Modelica.Units.SI.PerUnit Vsp = 1.0;
8  equation
9    S.re = Psp/data.Sbase;
10   Vsp = V;
11 end PVSource;
```

(d) `PQSource` model.

```
3  model PQSource
4    extends Interfaces.Partial_Source;
5    parameter Units.ActivePower Psp;
6    parameter Units.ReactivePower Qsp;
7  equation
8    S.re = Psp/data.Sbase;
9    S.im = Qsp/data.Sbase;
10 end PQSource;
```

This omission can be explained by the fact that the injections of reactive power at PV and Vθ buses can be obtained directly through the nodal reactive power equations, assuming that all voltage magnitudes and phases in the system are known.

Reactive power generated by generators or synchronous condensers is calculated after the operational state of the network is computed. Therefore, the traditional method for considering reactive power generation limits involves comparing the generated reactive power to its previously defined limits through the machine's capability curve after achieving convergence or partial convergence in the problem. If the generated reactive power exceeds its limits, the bus is transformed from a PV (voltage-controlled) into a PQ (specified reactive power) bus, where the specified reactive power value is set to the violated limit. Then, the iterative process is resumed.

In the traditional methodology of the power flow problem, this procedure is performed alternately with the iterative process [15]. An alternative way to perform this process is reported in [12], where the equations that model the treatment of limits

Code 2: Load model hierarchy.

(a) `ZIPLoad` model.

```
model ZIPLoad
  extends Interfaces.Partial_Load;
  parameter Modelica.Units.SI.PerUnit Vdef = 1.0;
  parameter Interfaces.LoadData ss_par =
    ↪ Interfaces.LoadData();
protected
  parameter Real pp = 1 - ss_par.pi - ss_par.pz;
  parameter Real pi = ss_par.pi;
  parameter Real pz = ss_par.pz;
  parameter Real qq = 1 - ss_par.qi - ss_par.qz;
  parameter Real qi = ss_par.qi;
  parameter Real qz = ss_par.qz;
equation
  S.re = (Psp+dpsp)/data.Sbase*(pp + pi*(V/Vdef) +
    ↪ pz*(V/Vdef)^2);
  S.im = (Qsp+dqsp)/data.Sbase*(qq + qi*(V/Vdef) +
    ↪ qz*(V/Vdef)^2);
end ZIPLoad;
```

(b) `Partial_Load` abstract model.

```
model Partial_Load
  outer SystemData data;
  extends Circuit.Interfaces.ShuntComponent;
  import Modelica.ComplexMath.conj;
  import Abs=Modelica.ComplexMath.abs;
  parameter Units.ActivePower Psp;
  parameter Units.ReactivePower Qsp;
  Modelica.Units.SI.ComplexPerUnit S;
  Modelica.Units.SI.PerUnit V(start = 1);

  Modelica.Blocks.Interfaces.RealInput dPsp if
    ↪ useExternalPsp;
  Modelica.Blocks.Interfaces.RealInput dQsp if
    ↪ useExternalQsp;
  parameter Boolean useExternalPsp = false;
  parameter Boolean useExternalQsp = false;
  Modelica.Blocks.Interfaces.RealOutput dpsp, dqsp;
equation
  S = v*conj(i);
  V = Abs(v);

  if useExternalPsp then
    connect(dPsp, dpsp);
  else
    dpsp = 0;
  end if;
  if useExternalQsp then
    connect(dQsp, dqsp);
  else
    dqsp = 0;
  end if;
end Partial_Load;
```

are incorporated into Newton's method without the need for external modifications in the iterative solution process. This modeling strategy for handling reactive power limits is one of the adopted in the OmniPES library for steady-state analysis.

In the proposed methodology for representing reactive limits at PV sources and Vθ sources, sigmoid-based switches are utilized. These switches transition between the values zero and one according to the evolution of the solution process. In summary, the methodology for handling reactive limits combines all equations of PV sources and Vθ sources into a single equation, describing the behavior of generators in both normal operating conditions (while adhering to their reactive power generation limits) and in conditions where these limits are exceeded. The parts of this equation are activated or deactivated through the sigmoid-based switches as the source's operating mode changes.

TABLE I: `PVSource_Qlim` operation modes map.

Operation Mode	Sigmoid Switches				Active Equation
	`ch1`	`ch2`	`ch3`	`ch4`	
Normal	0	0	–	–	$V_k - V_k^{sch}$
Superior Reactive Limit	1	0	1	–	$Q_{G_k} - Q_{G_k}^{max}$
Inferior Reactive Limit	0	1	–	1	$Q_{G_k} - Q_{G_k}^{min}$
Superior Limit Backoff	1	0	0	–	$V_k - V_k^{sch}$
Inferior Limit Backoff	0	1	–	0	$V_k - V_k^{sch}$

The equation (3) describes the steady-state behavior of a generator connected to a generic bus k. In this equation, it is possible to observe the behavior of the bus as a PV bus when generation limits are not reached, and the behavior as a PQ bus when one of its limits is violated. Note that V_k^{sch} denotes scheduled generator voltage magnitude, V_k is the actual generator voltage magnitude, while $Q_{G_k}^{max}$ and $Q_{G_k}^{min}$ are the maximum and minimum reactive power generation limits, respectively. Switch 1 (`ch1`) represents the generator identification reaching its maximum reactive power generation limit, while Switch 2 (`ch2`) identifies its lower limit. The backoff procedure, traditionally used in power flow programs, which is carried out by voltage behavior, is handled through switches 3 and 4 (`ch3` and `ch4`). Switch `ch3` identifies whether the bus voltage is above the scheduled value, and `ch4` whether the bus voltage is below the scheduled value.

$$\begin{aligned}(1-\texttt{ch1}\cdot\texttt{ch3})(1-\texttt{ch2}\cdot\texttt{ch4})(\;V_k - V_k^{sch}\;)+\\ +(\quad\texttt{ch1}\cdot\texttt{ch3})(1-\texttt{ch2}\cdot\texttt{ch4})(Q_{G_k}-Q_{G_k}^{max})+\\ +(1-\texttt{ch1}\cdot\texttt{ch3})(\quad\texttt{ch2}\cdot\texttt{ch4})(Q_{G_k}-Q_{G_k}^{min})=0\end{aligned} \tag{3}$$

Table I shows the mapping of sigmoid-based switches values in both normal operating mode and exceeded reactive limits mode. More specifically, the switches `ch1`, `ch2`, `ch3`, and `ch4` are implemented in the `algorithm` section shown in Code 3.

B.

Steady-State Results

To showcase the results obtained with the OmniPES library, the steady-state performance of two voltage control alternatives are compared. The first alternative considers the system operating according to traditional load-flow restrictions where generators control their respective terminal voltage magnitudes. In the second alternative, SVR is employed with bus #30 as the pilot node, while the generators' reactive powers are kept proportional to their respective rated apparent power.

In Fig. 2, generator G1 uses the `VTHSource_Qlim` model (reference bus with reactive power limits), while G2 uses the `PVSource_Qlim` model. The load connected to bus #30 utilizes the `Ctrl_ZIPLoad` model discussed in section IV. Both G2 and the load connected to bus #30 have inputs for an active power ramp that represents a load increase, accompanied by the corresponding generation redispatch. Generator G2 assumes 3/4 of the load ramp, while G1 assumes the remaining 1/4. Reactive power is also divided in the ratio of 1/4 and 3/4

Code 3: `Partial_VSource_Qlim_sigmoid` model.

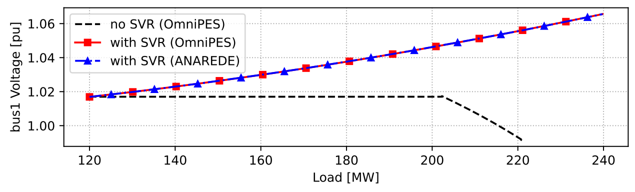
```
partial model Partial_VSource_Qlim_sigmoid
  outer SystemData data;
  extends Icons.Vsource;
  extends Interfaces.Partial_VSource_Qlim;
  parameter Modelica.Units.SI.PerUnit growth_rate = 1e5;
  parameter Modelica.Units.SI.PerUnit tolq = 1e-3;
  parameter Modelica.Units.SI.PerUnit tolv = 1e-3;

  Real ch1, ch2, ch3, ch4;
  protected
  final parameter Real lim_max = Qmax/data.Sbase - tolq;
  final parameter Real lim_min = Qmin/data.Sbase + tolq;
  Real lim_sup;
  Real lim_inf;
  (auxiliary code)
equation
  lim_sup = Vsp + dvsp + tolv;
  lim_inf = Vsp + dvsp - tolv;
  (1 - ch1*ch3)*(1 - ch2*ch4)*(V - Vsp - dvsp) +
  ↪  ch1*ch3*(1 - ch2*ch4)*(S.im - Qmax/data.Sbase) +
  ↪  (1 - ch1*ch3)*(ch2*ch4)*(S.im - Qmin/data.Sbase) =
  ↪  0;
algorithm
  ch1 := 1/(1 + exp(-growth_rate*(S.im - lim_max)));
  ch2 := 1/(1 + exp( growth_rate*(S.im - lim_min)));
  ch3 := 1/(1 + exp( growth_rate*(   V - lim_sup)));
  ch4 := 1/(1 + exp(-growth_rate*(   V - lim_inf)));
end Partial_VSource_Qlim_sigmoid;
```

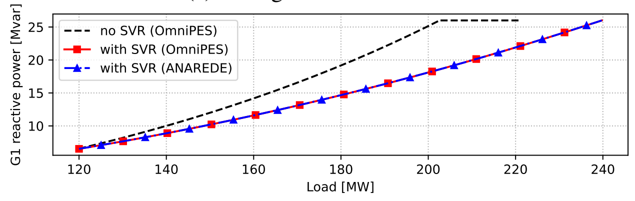
between generators G1 and G2, respectively, when the SVR is active.

The power flow results for all load ramp points are displayed in Fig. 3, which precisely align with those obtained in [10], [11]. The dynamic behavior of the generator G1 and the pilot bus is shown in figures 3(a), 3(b) and 3(c). In Fig. 3(b), the generator G1, without SVR, reaches its maximum reactive power limit at 26 Mvar when the active load ramp exceeds 200 MW. At this point, a noticeable change occurs in the sigmoid-based switches, as seen in Fig. 4, which forces the generator G1 to transition from voltage control to reactive power control at its maximum limit of 26 Mvar. In particular, `ch1` switches from zero to one, and `ch4` switches from one to zero, while `ch2` and `ch3` remain at zero and one, respectively. This transition corresponds to generator G1 shifting from Normal mode to Superior Reactive Limit mode, as defined in Table I. The previous operating mode change is also noticeable in 3(a), characterized by a voltage drop at the bus #1, the generator G1 terminal.

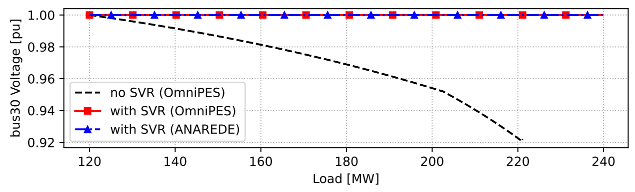
Also, in Fig. 3, results obtained with the OmniPES library on the OpenModelica simulation platform are compared to those obtained with the industrial-grade power flow program ANAREDE, developed by CEPEL (Brazil). Although visually both results are very coherent with each other, the Coefficient of the Variation of the Root Mean Square Error (CVRMSE) was also calculated. The CVRMSE is defined in (4) where $\mathbf{y}$ corresponds to a vector with the calculated variable, and $\mathbf{y}^*$ represents another vector with reference values. In the present comparisons, the reference values are those obtained with ANAREDE. Moreover, $\mathrm{E}(\mathbf{y}^*)$ returns the average value of the reference variable, which normalizes the root mean square error calculated with n computed and reference values.



(a) Voltage at G1 terminal.



(b) Reactive power in G1.



(c) Voltage at the load.

Fig. 3: Steady-state simulations with and without SVR.

TABLE II: Error metrics for the selected steady-state results.

Variable	CVRMSE [%]	MRE [%]	Load at MRE [MW]
G1 voltage	0.0026	0.0043	144
G2 voltage	0.0025	0.0044	168
G1 reactive power	0.0002	0.0003	228
G2 reactive power	0.0001	0.0002	228
Load voltage	0.0000	0.0000	120

$$\mathrm{CVRMSE}(\mathbf{y},\mathbf{y}^*) = \frac{1}{\mathrm{E}(\mathbf{y}^*)}\sqrt{\sum_{k=1}^{n}\frac{(y_k - y_k^*)^2}{n}} \tag{4}$$

The results for the CVRMSE calculated for the variables depicted in Fig. 3 are shown in Table II. One can observe that the computed CVRMSE for the voltages and reactive power at the generators remain lower than 0.003 %. Additionally, the Maximum Relative Error (MRE) values, followed by the load values they occur in, remain lower than 0.005 %.

V. Dynamic Analysis

For the dynamic analysis example, the generation part of the steady-state model of the tutorial system shown in Fig. 2 was modified, as shown in system detail depicted in Fig. 5. In this new system, the network and load remain unchanged, while the sources are replaced with synchronous machine-based hydroelectric power plants. In the sequence, the basic strategy for modeling both steady-state restrictions and dynamic behavior of the power plants will be discussed.

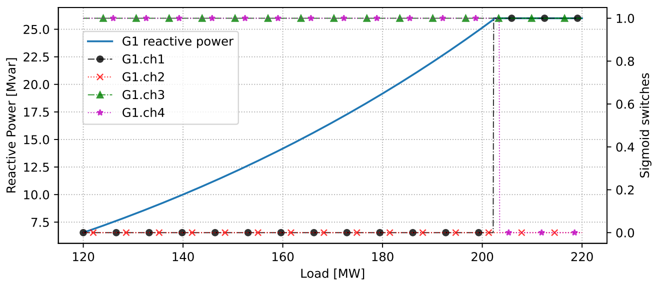


Fig. 4: Sigmoid-based switches and reactive power for G1.

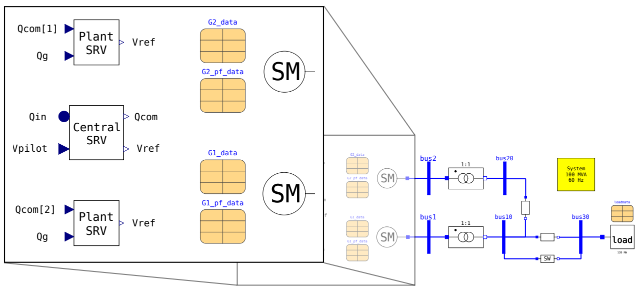


Fig. 5: Modifications for the dynamic analysis.

A. Embedded Power-Flow Restrictions

Every power system dynamic simulation requires a set of valid initial conditions, defined in accordance to previously established steady-state restrictions. Therefore, the same concepts discussed in the section IV need to be replicated in the dynamic models, but only for the initialization process of the simulation. In this manner, an abstract `Restriction` model, as shown in Code 4(b), serves as the base class for establishing other steady-state restrictions, such as the `Restriction_PV` model, which is detailed in Code 4(c).

The abstract `Restriction` model, shown in Code 4(b), does not implement any equations; instead, it declares four variables: active power `P`, reactive power `Q`, voltage magnitude `V` and voltage angle `theta`. The `initial equation` section is then implemented by derived models, as exemplified in the `Restriction_PV` model (Code 4(c)). In this model, both active power `P` and voltage magnitude `V` are set to their specified values, `Psp` and `Vsp`, respectively. These specified values are passed to restrictions through a record of type `RestrictionData`, show in Code 4(a). The same approach is employed for other restrictions, such as `Restriction_PQ`, shown in Code 4(d).

B. Power Plant Modeling

For the power plants, another abstract model was created, the `GenericSynchronousMachine` model, which is illustrated in Fig. 6. This model is composed of other sub-models properly interlinked. It's important to note that in this diagram, triangle-shaped arrows represent the unidirectional transmission of causal variables, while square-shaped connections exchange acausal variables, such as voltages and currents.

The electrical model characterizes the dynamics and constraints imposed by the electrical circuits within this machine

Code 4: Restriction model hierarchy.

(a) `Restriction` data.

```
record RestrictionData
  extends Modelica.Icons.Record;
  import OmniPES.Units;
  parameter Units.ActivePower Psp = 0.0;
  parameter Units.ReactivePower Qsp = 0.0;
  parameter Modelica.Units.SI.PerUnit Vsp = 1.0;
  parameter Modelica.Units.SI.Angle theta_sp = 0;
end RestrictionData;
```

(b) `Restriction` abstract model.

```
partial model Restriction
  outer SystemData data;
  parameter RestrictionData param;
  Modelica.Units.SI.PerUnit P;
  Modelica.Units.SI.PerUnit Q;
  Modelica.Units.SI.PerUnit V;
  Modelica.Units.SI.Angle theta;
end Restriction;
```

(c) `Restriction_PV` model.

```
model Restriction_PV
  extends Restriction;
initial equation
  P = param.Psp/data.Sbase;
  V = param.Vsp;
end Restriction_PV;
```

(d) `Restriction_PQ` model.

```
model Restriction_PQ
  extends Restriction;
initial equation
  P = param.Psp/data.Sbase;
  Q = param.Qsp/data.Sbase;
end Restriction_PQ;
```

and produces electrical active power `Pe` and its terminal voltage magnitude `Vt` as outputs. The `Inertia` model takes in electrical power `Pe`, converted by the generator, and mechanical power `Pm`, developed by the prime mover. It then outputs the angular speed and the load angle with respect to the arbitrary angular reference frame within the network.

The speed regulator (`SR` model) receives the generator's angular speed and delivers the mechanical power `Pm` developed by the prime mover. Meanwhile, the automatic voltage regulator (`AVR` model) takes in the generator's terminal voltage magnitude `Vt` and the stabilizing signal `Vsad`, produced by the power system stabilizer (`PSS` model). It's worth noting that, for the present simulations, no PSS model was necessary.

Except for the `Inertial` model, all submodels are declared as `partial` abstract models in the Modelica language. This declaration implies that actual implementations are required for each sub-model to complete the generator model. As an example, the `PartialElectrical` model is detailed in Code 5(a). In this model, only interface restrictions (lines 30-37) are described, along with stator voltage and flux linkage (lines 38-39) and the electrical converted power (line 40). The number of rotor circuits, however, depends on the generator type, whether it corresponds to a hydro or turbogenerator. Specialization is achieved with derived models, such as the `Model_2_1_Electric` model, partially shown in Code 5(b), which represents the rotor circuitry formed by

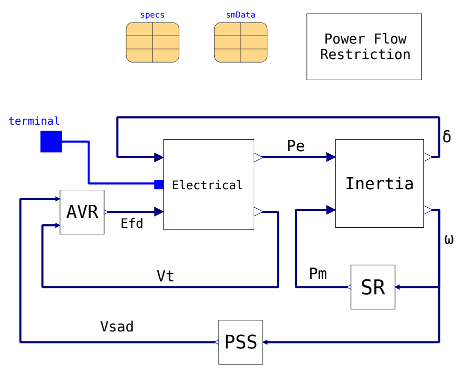


Fig. 6: `GenericSynchronousMachine` model.

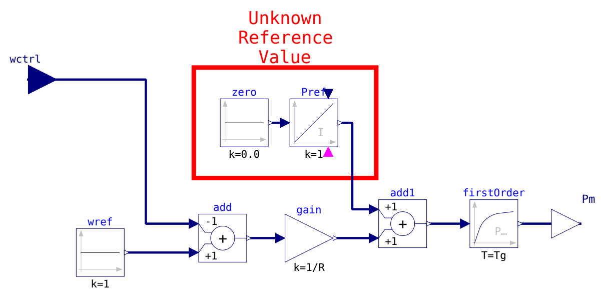


Fig. 7: `SpeedGovernor` model.

a field winding and two damper windings without saturation effects [16]. The naming convention for the electrical models follows the one established in [17].

The actual implementations for the speed regulator is depicted in Fig. 7, for illustration purposes. The speed regulator employs a proportional controller with a droop gain denoted by `R`, along with a first-order transfer function representing the time-lag imposed by the prime mover.

The voltage regulator is based on the AC4C-type excitation system, as reported in [18]. It is modeled as a single time constant block with non-windup limits [16]. Detailed descriptions of all models can be found in [10], [11].

It's important to note that the inheritance mechanism, intrinsic to the Modelica language, enables rapid model switching using the `redeclare` statement. This means that any derived model of a parent can be utilized to implement a specific block.

The dynamic models associated with the power plants also rely on reference values for the controllers, which are inevitably linked to the operating conditions defined by the power-flow restrictions. Take, for example, the speed regulator illustrated in Fig. 7, which requires two reference values: one for angular speed and another for active power. The former was set to unity, while the latter needs to be calculated to ensure that the error passed as an input to the first-order block (representing the turbine) is zero in steady-state. In Modelica, an unknown constant variable can be represented by setting its derivative to zero in steady-state, a condition achieved by using an integrator that receives a constant zero input. This way, the state of the

Code 5: Machines electrical model hierarchy.

(a) `PartialElectrical` model.

```
initial equation
  der(delta) = 0.0;
equation
  Vabs = abs(terminal.v);
  theta = arg(terminal.v);
  Vqd = sys2qd(terminal.v, delta);
  Iqd = sys2qd(-terminal.i, delta);
  St = Vqd*conj(Iqd);
  Pt = St.re;
  Qt = St.im;
  Vt = Vabs;
  Vqd = -ra*Iqd - j*Fqd;
  Fqd = xl*Iqd + Faqd;
  Pe = Fqd.im*Iqd.re - Fqd.re*Iqd.im;
```

(b) `Model_2_1_Electric` model.

```
initial equation
  der(F1d) = 0;
  der(Fkd) = 0;
  der(Fkq) = 0;
equation
  T1d0*der(F1d) = Efd - (xd - xl)*Ifd;
  T2d0*der(Fkd) = (x1d - xl)^2/(x2d - x1d)*Ikd;
  T2q0*der(Fkq) = (xq - xl)^2/(x2q - xq)*Ikq;
  Ifd = ((x1d - xd)/(xd - xl)*Faqd.im + F1d)/(x1d - xl);
  Ikd = (x2d - x1d)*(Faqd.im - Fkd)/(x2d - xl)/(x1d -
  ↪  xl);
  Ikq = (x2q - xq)*(Faqd.re - Fkq)/(x2q - xl)/(xq - xl);
  Faqd.im = (x2d - xl)*Iqd.im + (x2d - xl)*F1d/(x1d -
  ↪  xl) + (x1d - x2d)*Fkd/(x1d - xl);
  Faqd.re = (x2q - xl)*Iqd.re + (xq - x2q)/(xq - xl)*Fkq;
```

integrator associated with the active power reference value can be solved concurrently with the power-flow restrictions.

C. *Secondary Voltage Regulation*

The tutorial system, as depicted in Fig. 5, was designed to illustrate the dynamics of an SVR scheme, as described in [10], [11]. In this scheme, a set of voltage control devices, including power plants, static compensators, synchronous condensers, LTC transformers, and others, can be coordinated to effectively regulate voltage at specific buses within an electric network. The primary objective of SVR schemes is to enhance system security by automatically improving voltage profiles during system operation. Typically, the time window for SVR dynamics is on the order of minutes.

The SVR control loop is implemented using external control blocks. The local component of the rfull is realized within the `Plant_SVR` model, utilizing standard Modelica models such as `Add` and `PI` controller blocks. This portion of the SVR control loop regulates the reactive power produced by each generator, resulting in the signal `Vref`, which is added to the voltage reference of that specific generator. Each generator participating in the SVR scheme is associated with a distinct instance of the `Plant_SVR` model.

The global portion of the SVR takes inputs from the reactive power generated by every participating generator, represented as the array `Qin`, and the voltage magnitude of the pilot bus, `Vpilot`, which requires control. Using these reactive powers, the central SVR controller calculates new commanded reactive powers (`Qcom[1]` and `Qcom[2]`) while maintaining the total reactive power produced by the generators.

Code 6: Interconnection among plant and central SVR controllers.

The interconnection among SVR-related controllers, generators, and the pilot bus is illustrated in Code 6. Here, one can notice that each plant's AVR reference is determined by the sum of the reference voltage computed by `central_SVR` and a locally generated voltage reference (lines 28 and 29). The reactive power produced by each plant is transmitted to their respective local SVR controllers, `g1_svr` and `g2_svr`, as well as to the central SVR controller, `central_SVR` (lines 30 amd 31). The reactive power reference for each plant, defined by `central_SVR`, is then conveyed to `g1_svr` and `g2_svr` (line 33). Finally, the voltage magnitude of the pilot bus is set equal to that of `bus30` in line 34.

Steady-state conditions for the tutorial system with secondary voltage regulation require a few adjustments, easily implemented using the Modelica language and its inheritance mechanism. Firstly, the central SVR imposes that the voltage magnitude at the load bus must be fixed. Consequently, at the load bus, the voltage magnitude, active power, and reactive power are known, leaving only the voltage phase as unknown. Thus, with three out of four variables of a single electrical node known, the voltage magnitude at the terminal of one of the power plants is required to remain unknown, dependent on the voltage magnitude at the load.

Additionally, the central SVR also requires that the reactive power output of each power plant must be proportional to its rated apparent power. Consequently, the voltage magnitude in a second power plant is treated as an unknown. In summary, in terms of power-flow nomenclature, the load bus is transformed into a `PQV`-bus, and the reactive power output of one power plant is made dependent on the other. As a result, the power plants' buses become `P` and `θ` buses. This transformation is necessary due to the need for an angular reference in the system.

D. Simulation Results

The dynamic behavior of the tutorial system began from a steady-state condition, in which the voltage at bus 30 was set to 1 pu. The distribution of reactive power for the power plants was configured so that power plant `G1`, connected to bus 1, produced one-third of the reactive power generated by power plant `G2` [10], [11].

For the sake of accuracy checking, obtained results are also compared to those produced by the industrial-grade transient stability program ANATEM, also developed by CEPEL.

For the initial condition calculations, power plant `G2` defines its dispatched active power, while `G1` establishes the angular reference. During the voltage reference calculations, either of the plant SVRs, `g1_svr` or `g2_svr`, may explicitly set the derivative of `Vref` to zero, but not both simultaneously.

The results for the initial conditions of the tutorial system are presented in Table III. It is evident that at the beginning of the simulation with the SVR, the pilot bus voltage (`bus30`) was successfully set to unity while maintaining the one-third ratio for the reactive power generated by `G1` and `G2`. However, in the original operating conditions proposed for the tutorial system in [10], without the SVR, achieving these goals required manual adjustments only at the beginning of the simulation, as indicated by the similarity of the values in both columns in Table III.

TABLE III: Initial values for selected variables in pu.

Variables	**No SVR**	**With SVR**
`bus1.V`	1.01700	1.01691
`bus2.V`	1.02500	1.02500
`bus10.V`	1.00586	1.00582
`bus20.V`	1.01352	1.01350
`bus30.V`	1.00004	1.00000
`G1.electrical.Pt`	0.30000	0.30000
`G2.electrical.Pt`	0.90000	0.90000
`G1.electrical.Qt`	0.06548	0.06522
`G2.electrical.Qt`	0.19537	0.19565
`g1_srv.Vref`	–	0.00000
`g2_srv.Vref`	–	0.00811
`central_SVR.Vref`	–	1.03108

TABLE IV: Error metrics for selected dynamic results.

Variable	CVRMSE [%]	MRE [%]	Instant at MRE [s]
Load Voltage	0.0015	0.0108	27.1660
G1 reactive power	0.4227	2.3054	28.3660
G2 reactive power	0.1404	0.8249	28.3700

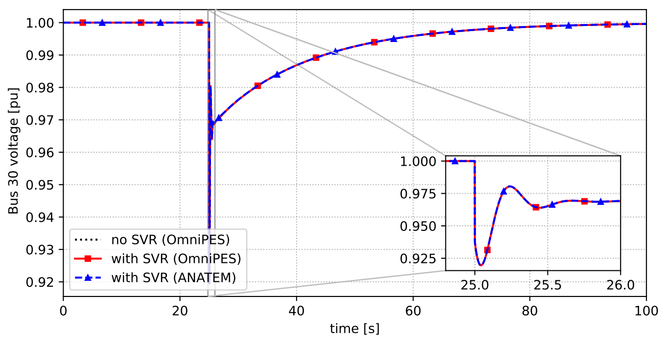
The tested contingency involved opening one of the transmission lines after 25 s of steady-state operation. As depicted in Fig. 8(a), the system's SVR successfully regulates the voltage at the pilot bus. Without SVR, the voltage would tend to settle at a significantly lower magnitude. Additionally, the power plants are required to produce less reactive power when compared to the system without SVR. This reduction is mainly due to the decreased reactive power produced by the power plant G1 under SVR control.

The curves depicted in Fig. 8 exhibit acceptable coherence between OmniPES and ANATEM. This observation is further reinforced by the error metrics (see subsection IV-B for metrics definitions) reported in Table IV. Notably, the CVRMSE for the load voltage is lower than 0.002 %, while for the reactive powers, it remains below 0.5 %. Among the variables observed, the maximum relative error (MRE) of 2.3 % is recorded for the reactive power at generator G1 at 28.37 s.

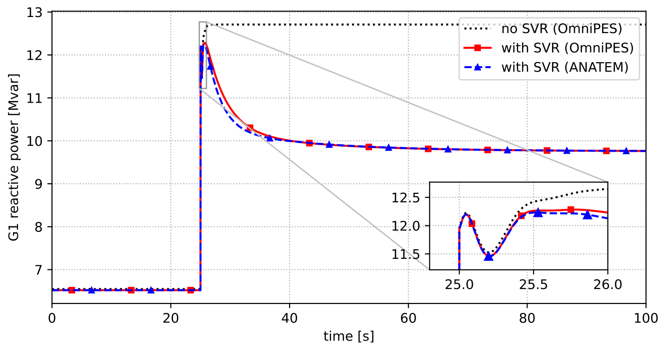
VI. Conclusion

In this paper, the Modelica library, OmniPES, is presented. This library is designed for rapid prototyping of small to medium-scale power systems, facilitating both steady-state and transient analysis. The primary objective was to align the library closely with the modeling strategies employed by production-grade power flow and transient stability programs.

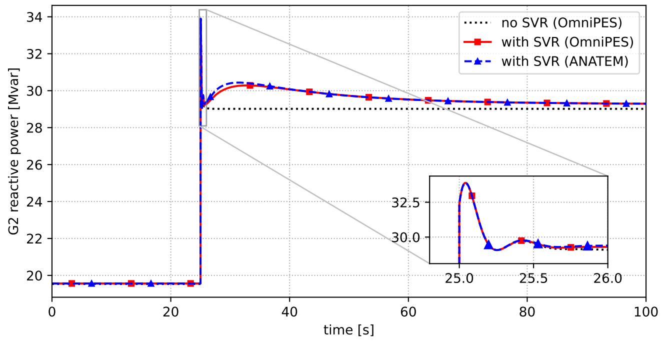
Among the notable features of this library, the modeling of generators' reactive power limits using sigmoid-based switches can be highlighted. These switches allow for accurate



(a) Voltage magnitude at the pilot bus.



(b) G1 reactive power.



(c) G2 reactive power.

Fig. 8: Selected dynamic simulation results.

constraints on reactive power production while adjusting controlled variables as required by the system solution. It is important to note that the sigmoid-based switches also permit the system to backoff from its limits as the system operating conditions allow.

Another noteworthy feature pertains to the initial conditions for transient simulations. These conditions are implicitly defined by embedded power-flow restrictions. Consequently, users are not obligated to import power flow results from external third-party tools.

The implementation in the Modelica language has been thoroughly tested using a small-scale system to evaluate a secondary voltage control scheme. The presented results validate and clearly demonstrate the accuracy and effectiveness of this proposed library for modeling and simulating both traditional and more complex system scenarios.

Acknowledgment

The authors would like to thank the National Research Council (CNPq), the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES), the State of Minas Gerais Research Foundation (FAPEMIG), and INERGE.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI's ChatGPT 3.5 in order to proofread the final text. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

[1] Supporting LAC countries in their reform efforts. International Energy Agency, (IEA). [Online]. Available: https://www.oecd.org/latin-america/countries/

[2] Monthly electricity statistics. International Energy Agency, (IEA). [Online]. Available: https://www.iea.org/data-and-statistics/data-product/monthly-electricity-statistics

[3] The Modelica Association. Modelica language. [Online]. Available: https://www.modelica.org/

[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, Jan. 2016, publisher: Elsevier.

[5] M. de Castro, D. Winkler, G. Laera, L. Vanfretti, S. A. Dorado-Rojas, T. Rabuzin, B. Mukherjee, and M. Navarro, "Version [OpenIPSL 2.0.0] - [iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations]," vol. 21, p. 101277.

[6] A. Bartolini, F. Casella, and A. Guironnet, "Towards Pan-European Power Grid Modelling in Modelica: Design Principles and a Prototype for a Reference Power System Library," in *Linköping Electronic Conference Proceedings*. Linköping University Electronic Press.

[7] R. Franke and H. Wiesmann, "Flexible modeling of electrical power systems – the modelica PowerSystems library," in *Linköping Electronic Conference Proceedings*. Linköping University Electronic Press.

[8] Modelica Association, "Modelica Standard Library (MSL)," github repository. [Online]. Available: https://github.com/modelica/ModelicaStandardLibrary

[9] M. A. Tomim, R. M. Henriques, and J. A. P. Filho, "OmniPES: Modelica Library for Power Systems Simulation," Github repository. [Online]. Available: https://github.com/marcelotomim/OmniPES

[10] N. Martins, J. Ferraz, S. Gomes, P. Quintao, and J. Passos, "A Demonstration Example of Secondary Voltage Regulation: Dynamic Simulation and Continuation Power Flow Results," in *2001 Power Engineering Society Summer Meeting. Conference Proceedings (Cat. No.01CH37262)*, vol. 2, pp. 791–796 vol.2.

[11] N. Martins, S. Corsi, C. Taylor, C. Vournas, C. Canizares, G. N. Taranto, G. Fabrice, H. Lefebvre, J. Paserba, J. V. Hecke, J. Sanchez-Gasca, J. C. R. Ferraz, K. Uhlen, L. Pereira, M. Pozzi, M. C. Perdomo, N. Miller, P. Kundur, V. C. Thierry, and V. Ajjarapu, "Coordinated Voltage Control in Transmission Networks," p. 285, task Force C4.602.

[12] J. P. Peters Barbosa and J. Alberto Passos Filho, "New methodologies for SVC modeling in the power flow problem based on sigmoid functions," *Electrical Engineering*, 2023.

[13] T. Van Cutsem and C. Vournas, *Voltage Stability of Electric Power Systems*. Springer Science+Business Media Dordrecht.

[14] The Modelica Association, "OpenModelica." [Online]. Available: https://openmodelica.org/

[15] B. Stott, "Review of load-flow calculation methods," *Proceedings of the IEEE*, vol. 62, no. 7, pp. 916–929, 1974.

[16] P. Kundur, *Power System Stability and Control*. McGraw-Hill.

[17] "IEEE Guide for Synchronous Generator Modeling Practices and Parameter Verification with Applications in Power System Stability Analyses," *IEEE Std 1110-2019 (Revision of IEEE Std 1110-2002)*, pp. 1–92, 2020.

[18] "IEEE Recommended Practice for Excitation System Models for Power System Stability Studies," pp. 1–207.