Experience on the use of the DAE mode in industrial power system simulations

OpenModelica Workshop – Linköping – 03/02/2020

K. Abdelhak, B. Bachmann (FH Bielefeld) and F. Rosière, A. Guironnet (RTE)

Outline

- Introduction to power system simulations
- OpenModelica DAE mode
- Examples of DAE mode advantages
- Conclusion

01

Introduction to power system simulations

Transmission System Operators

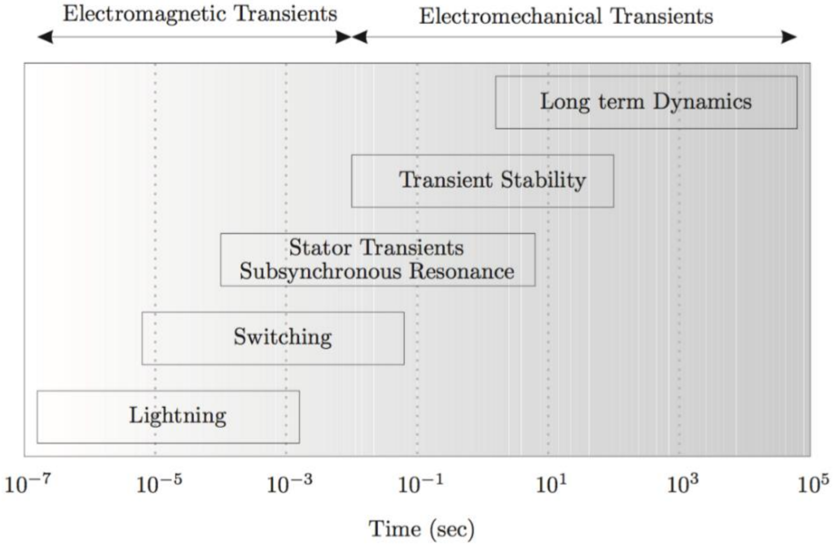
- "Entities operating independently from the other electricity market players and responsible for the bulk transmission of electric power on the main high voltage electric network"
  - ➢ Non discriminatory and transparent access to the electricity grid
  - ➢ Safe operation and maintenance of the system
  - ➢ Grid infrastructure development

- RTE – French Transmission System Operator
  - ➢ In charge of the largest European network (100 000 kms of EHV and HV lines – 400 to 63 kV, 2 600 substations, peak load served > 100 GW).
  - ➢ Ensuring a stable and secure grid operation means:
    - ❖ Adequacy – Acceptable steady-state (thermal overload, voltage values for materials)
    - ❖ Stability – Stable and possible transition between different operating points

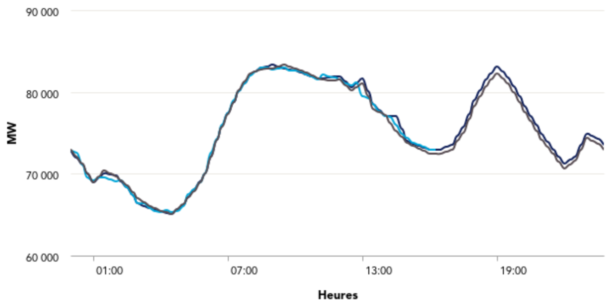
      Dynamic stability (transient, voltage, small-signal, frequency, etc.) ensured by time-domain simulations

Time-domain simulations

- Analysis of the system evolution during transitions
  - Triggered by the normal evolution of the system (load change, production scheduled change, etc.) or by sudden change (generator tripping, short-circuit, etc.)
  - Refer to a large range of phenomena with different time constants

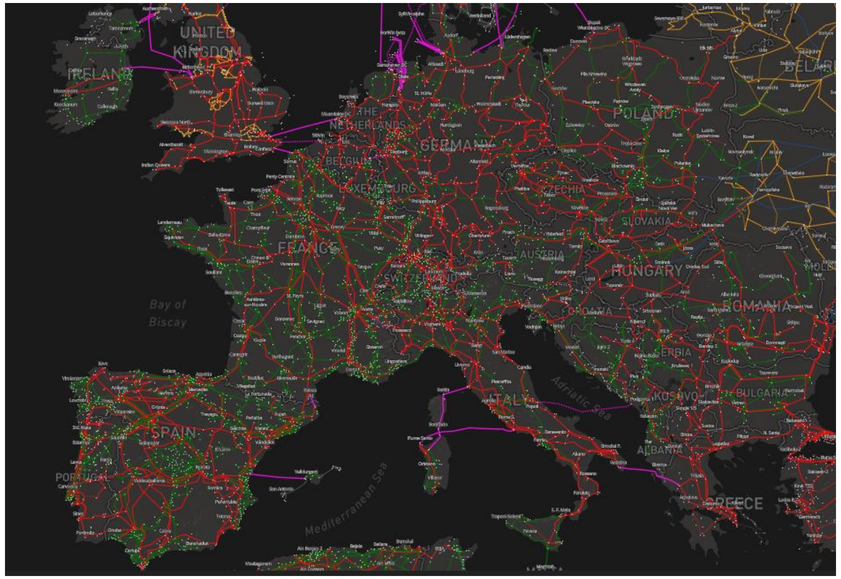


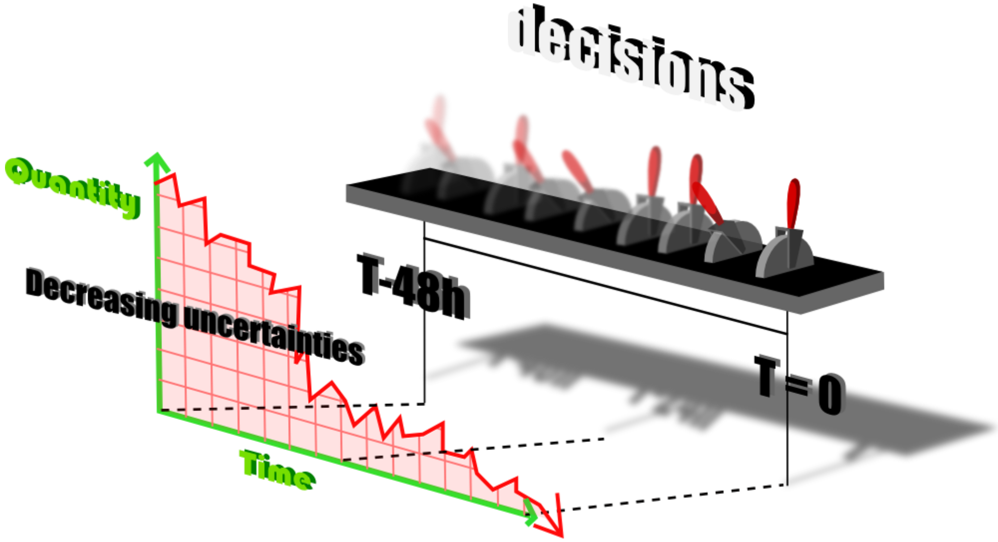
- Two main domains:
  - Electromagnetic transients (known as EMT):
    - Time constants from 1 ns to 1 ms
    - No dynamics neglegted
    - All the components have differential equations
  - Electromechanical transients (known as TS):
    - Time constants from 1 ms to several minutes
    - Fast dynamics (in particular in the network) are neglegted
    - Phasor approximation, no dynamic in the network



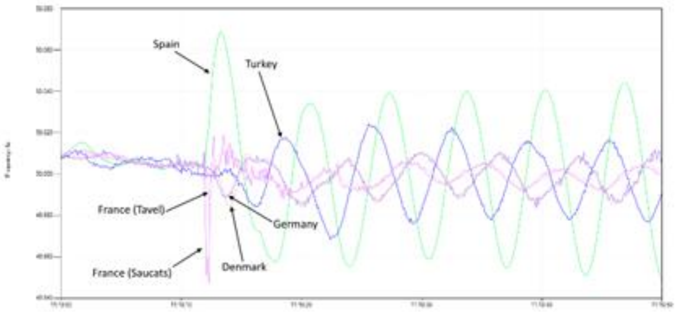
Large-scale industrial simulations

- Phasor or TS simulations are done frequently and on large scale networks
  - ➢ Voltage and transient stability studies are run automatically with real time data and hours, days and week-aheads on different scenarios
  - ➢ Dynamic security assessment: simulate all network contingency every 15’
  - ➢ Switch from a physically-driven network to a software-based network will even reinforce the pressure on the simulations to be done.





Large-scale industrial simulations



- Large-scale phasor simulations complexity
  - **Spatial**: from regional to panEuropean studies (interarea oscillations) (10 000 electrical nodes, 3 000 generators -> 130 000 variables)
  - **Temporal**: from electromechanical phenomena (~1 ms) to slow dynamics (secondary voltage regulation – minutes) -> Stiff problems
  - **Hybrid**: discontinuities (tap changer change in a transformer, short circuit, etc.)

$\Rightarrow$Large set of hybrid sparse stiff semi-explicit index 1 DAE system

$$\begin{cases} t_0 \le t \le t_c \begin{cases} \dot{x}_t = f_a(x_t, y_t) \\ g_a(x_t, y_t) = 0 \end{cases} \\ C_{ab}(t_c, x_{t_c}, y_{t_c}) = 0 \\ t > t_c \begin{cases} \dot{x}_t = f_b(x_t, y_t) \\ g_b(x_t, y_t) = 0 \end{cases} \end{cases}$$

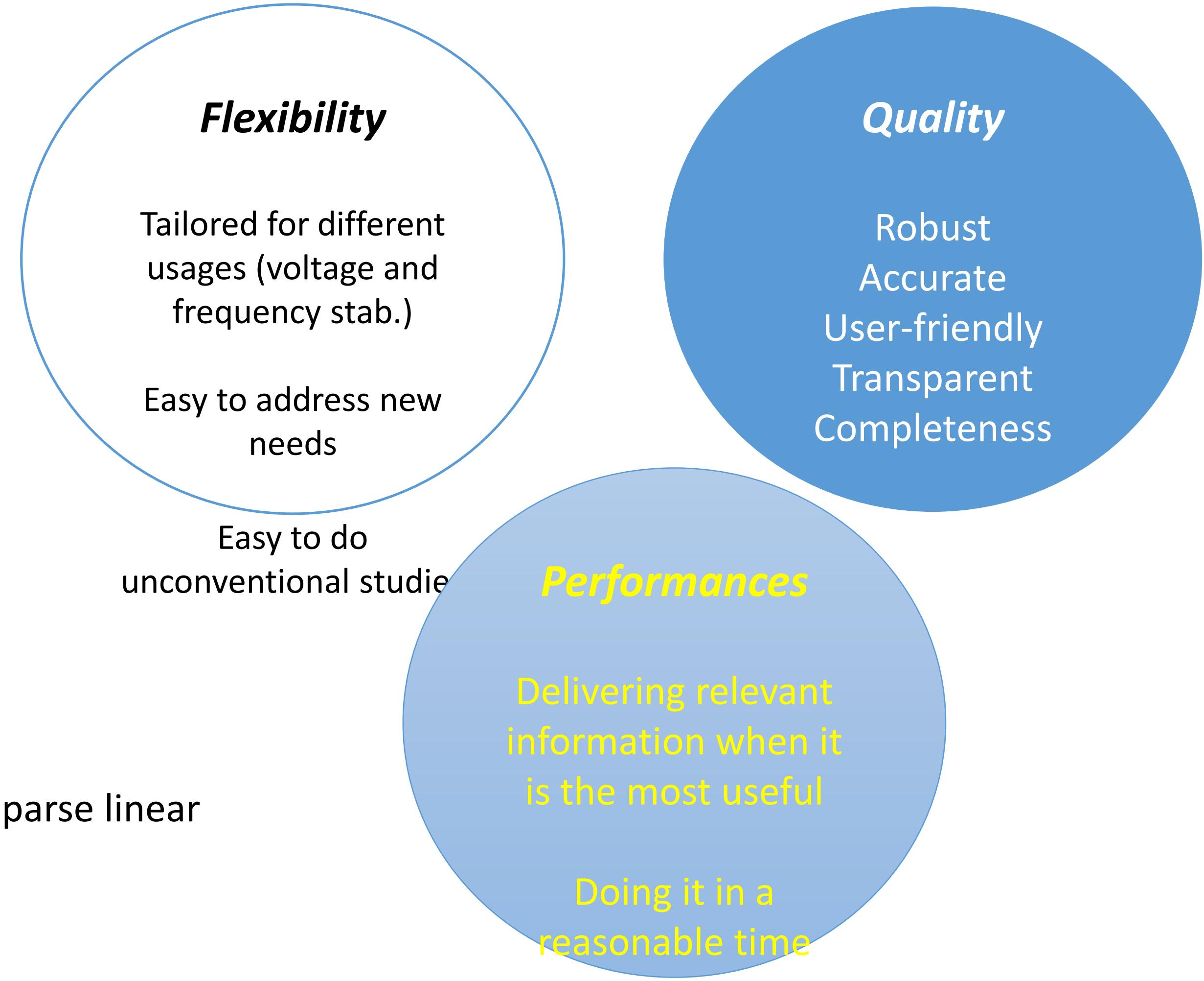
$$\begin{cases} f_a(x_{t_c}, y_{t_c}) \neq f_b(x_{t_c}, y_{t_c}) \\ g_a(x_{t_c}, y_{t_c}) \neq g_b(x_{t_c}, y_{t_c}) \end{cases}$$

Challenges and numerical methods

- Finding an acceptable enough trade-off between performance, flexibility and accuracy

- Numerical methods optimized for power system simulations
  - ➢ Taking advantage of the sparsity structure of the network
  - ➢ Sticking to an implicit DAE problem
  - ➢ Controlling accuracy

⇒ Variable time-step with implicit integration methods and sparse linear solvers are the reference for power system simulations



Flexibility

Tailored for different usages (voltage and frequency stab.)

Easy to address new needs

Easy to do unconventional studie

Quality

Robust
Accurate
User-friendly
Transparent
Completeness

Performances

Delivering relevant information when it is the most useful

Doing it in a reasonable time

Modelica-based simulations

- Modelica is promising for power system modelling and simulation
  - ➢ Models easy to write, share and understand
  - ➢ Generic and open source language
  - ➢ Adapted for physical, controls and even multi-system modelling.
  - ⇒ Gaining interest in the power system community and promoting by some actors

- Existing barriers or difficulties for operational large-scale simulations in Modelica with Modelica tools
  - ➢ Large system-wide centralized controls
  - ➢ Dynamic connectivity/topology analysis
  - ➢ Performances (runtime compilation and simulation time)
    - ❖ *Back to 2016*: Simulation time on IEEE57 75* slower than real time and compilation on larger networks fails or takes too much time[1]
    - ❖ Transformation to ODE and algebraic loops processing is one of the bottleneck

---

1. R. Viruez, S. Machado, L. M. Zamarreño, G. León, F. Beaude, S. Petitrenaud, and J.-B. Heyberger, “A Modelica-based Tool for Power System Dynamic Simulations,” Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017, Jul. 2017.

Modelica-based simulations

- Modelica is promising for power system modelling and simulation
  - ➢ Models easy to write, share and understand
  - ➢ Generic and open source language
  - ➢ Adapted for physical, controls and even multi-system modelling.
  - ⇒ Gaining interest in the power system community and promoting by some actors

- Domain-specific tools development enables to bypass some limitations
  - ➢ Hybrid C++ / Modelica simulation tool, initially developed by RTE – Dynaωo (http://dynawo.org)
  - ➢ Using native DAE sparse solvers formalism (breaking the LS and NLS built by OpenModelica Compiler)
  - ➢ A few tricks to avoid large algebraic loops (model by model compilation, C++ network)
  - ➢ Performances similar to current power system simulation tools
  - ➢ Come to us after the presentation if you want more details

Modelica-based simulations

- Modelica is promising for power system modelling and simulation
  - ➢ Models easy to write, share and understand
  - ➢ Generic and open source language
  - ➢ Adapted for physical, controls and even multi-system modelling.
  - ⇒Gaining interest in the power system community and promoting by some actors

⇒A strong need for a DAE mode in OpenModelica

- ❖ For enabling up to medium-size networks simulation in OM in the near future and envision large-size networks simulation as a possible target
- ❖ For making it possible for power system actors (in general) to work with Modelica environments

02

DAE mode in OpenModelica

Implementation Overview

Pipeline

Frontend

Backend

Code Generation

Simulation

Ideal DAE-Mode

- Skip the *Backend* entirely.
- Create residual DAE mode equations from *Frontend* structure.

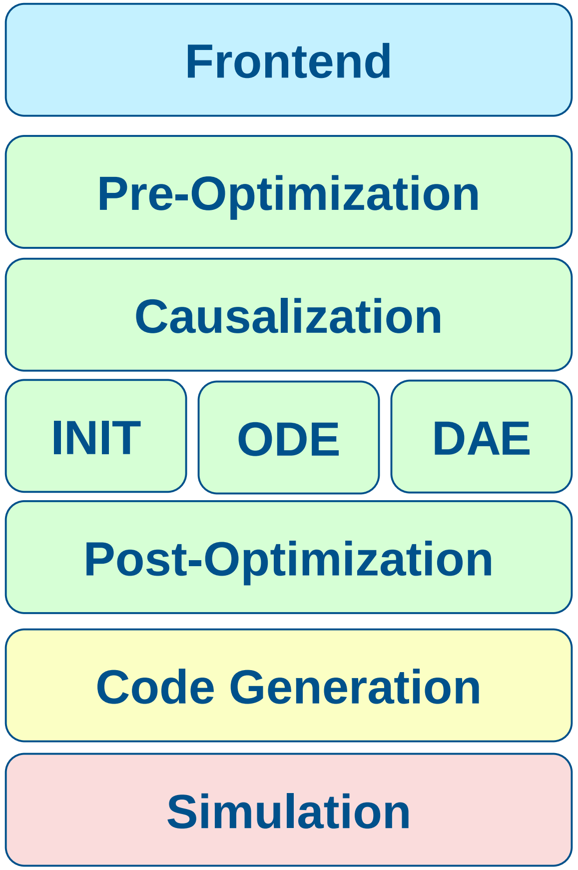
Problems

- Initialization
- Event Handling
- Index Reduction

FH Bielefeld
University of
Applied Sciences

Implementation Overview

Pipeline


Frontend

Pre-Optimization

Causalization

INIT | ODE | DAE

Post-Optimization

Code Generation

Simulation

Applicable DAE-Mode

Causalize and create following systems:
- Initialization (INIT)
- Event Handling (ODE)
- Simulation (DAE)

Advantages

- Tearing only affects INIT and ODE system
- Index Reduction is applied
- All Algebraic Loops are solved in one sparse system

Causalization

Initial System

$$0 = F(\dot{\underline{x}}(t), \underline{x}(t), \underline{y}(t), \underline{u}(t), \underline{q}(t), \underline{q}_{pre}(t), \underline{p}, t)$$

$$0 = \hat{F}(\dot{\underline{x}}(t_0), \underline{x}(t_0), \underline{y}(t_0), \underline{u}(t_0), \underline{q}(t_0), \underline{p}, t_0)$$

$$\dot{\underline{x}}(t_0) = \hat{f}(\underline{x}(t_0), \underline{u}(t_0), \underline{q}_{pre}(t_0), \underline{p}, t_0)$$

$$\underline{x}(t_0) = \hat{s}(\underline{x}(t_0), \underline{u}(t_0), \underline{q}_{pre}(t_0), \underline{p}, t_0)$$

$$\underline{y}(t_0) = \hat{g}(\underline{x}(t_0), \underline{u}(t_0), \underline{q}_{pre}(t_0), \underline{p}, t_0)$$

$$\underline{q}(t_0) = \hat{h}(\underline{x}(t_0), \underline{u}(t_0), \underline{q}_{pre}(t_0), \underline{p}, t_0)$$

INIT

$$\underline{z}(t_0) = \begin{pmatrix} \dot{\underline{x}}(t_0) \\ \underline{x}(t_0) \\ \underline{y}(t_0) \\ \underline{q}(t_0) \end{pmatrix} = \begin{pmatrix} \hat{f}(\,.\,) \\ \hat{s}(\,.\,) \\ \hat{g}(\,.\,) \\ \hat{h}(\,.\,) \end{pmatrix}$$

Variables	Description
$\dot{\underline{x}}(t)$	State Derivative
$\underline{x}(t)$	State
$\underline{y}(t)$	Algebraic Variables
$\underline{u}(t)$	Inputs
$\underline{q}(t)$	Discrete Variables
$\underline{q}_{pre}(t)$	Discrete Pre-Variables
$\underline{p}$	Parameters
t	Time

Causalization

Simulation System

$$0 = F(\dot{\underline{x}}(t), \underline{x}(t), \underline{y}(t), \underline{u}(t), \underline{q}(t_e), \underline{q}_{pre}(t_e), \underline{p}, t)$$

$$\underline{z}(t) = \tilde{F}(\,.\,) \begin{cases} \dot{\underline{x}}(t) = f(\underline{x}(t), \underline{u}(t), \underline{q}(t), \underline{p}, t) \\ \underline{y}(t) = g(\underline{x}(t), \underline{u}(t), \underline{q}(t), \underline{p}, t) \end{cases}$$

$$\underline{q}(t) = h(\underline{x}(t), \underline{u}(t), \underline{q}_{pre}(t), \underline{p}, t)$$

ODE

$$\underline{z}_E(t) = \begin{pmatrix} \dot{\underline{x}}(t) \\ \underline{y}(t) \\ \underline{q}(t) \end{pmatrix} = \begin{pmatrix} f(\,.\,) \\ g(\,.\,) \\ h(\,.\,) \end{pmatrix}$$

DAE

$$0 = \tilde{F}(\,.\,) - \underline{z}(t)$$

Variables	Description
$\dot{\underline{x}}(t)$	State Derivative
$\underline{x}(t)$	State
$\underline{y}(t)$	Algebraic Variables
$\underline{u}(t)$	Inputs
$\underline{q}(t_e)$	Discrete Variables
$\underline{q}_{pre}(t_e)$	Discrete Pre-Variables
$\underline{p}$	Parameters
t	Time

Updates

Bugfixes / Features

Support *Removed* Equations

- Equations without return value (e.g. asserts, dumping)
- Extra section outside simulation system

Advantage: Obvious ordering
Disadvantage: Forced ordering

Updated *Auxiliary Variable* Handling

- Variables generated from *Backend Modules* (e.g. *CSE*, when/if condition)
- Prevent implicit solving
- Moved to extra section

Advantage: Faster simulation
Disadvantage: Restrictions on Causalization

03

Example of DAE mode advantages

Compilation failure

- ODE compilation fails with a simplified PV generator model put on a simple network
  - Used for simplified time-domain simulations
  - When we don’t have enough data to represent the dynamic evolution of the generator

```
equation

  URefPu.value = UPu + LambdaPu * QGenRefPu;

  when QGenRefPu >= QMaxPu and pre(qStatus) <> QStatus.AbsorptionMax then
    qStatus = QStatus.AbsorptionMax;
    Timeline.logEvent1(TimelineKeys.GeneratorPVMaxQ);
  elsewhen QGenRefPu <= QMinPu and pre(qStatus) <> QStatus.GenerationMax then
    qStatus = QStatus.GenerationMax;
    Timeline.logEvent1(TimelineKeys.GeneratorPVMinQ);
  elsewhen (QGenRefPu < QMaxPu and pre(qStatus) == QStatus.AbsorptionMax) or (QGenRefPu > QMinPu and pre(qStatus) ==
QStatus.GenerationMax) then
    qStatus = QStatus.Standard;
    Timeline.logEvent1(TimelineKeys.GeneratorPVBackRegulation);
  end when;

  if running.value then
    QGenPu = if qStatus == QStatus.AbsorptionMax then QMaxPu else if qStatus == QStatus.GenerationMax then QMinPu else
QGenRefPu;
  else
    QGenPu = 0;
  end if;
```

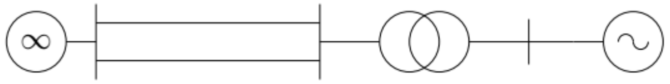


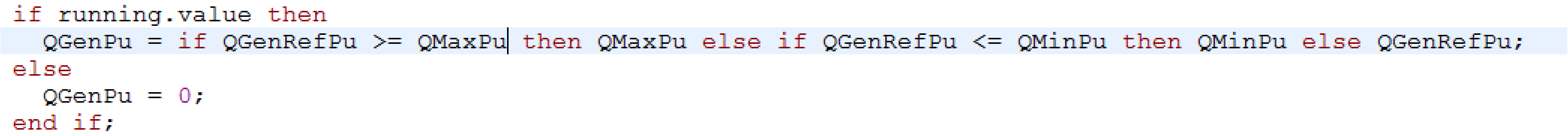
Figure 1: SMIB system representation

Compilation failure

- ODE compilation fails on a simplified PV generator model
  - ➢ Used for simplified time-domain simulations
  - ➢ When we don’t have enough data to represent the dynamic evolution of the generator

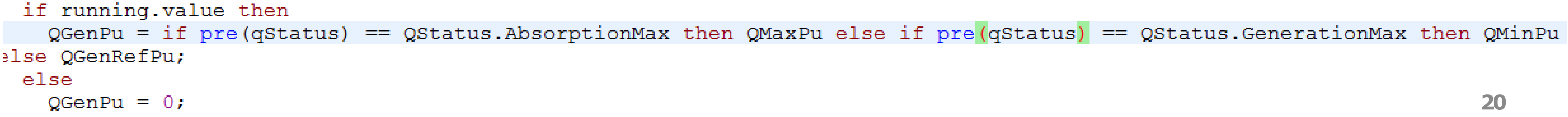
- DAE simulation works

- Writing differently the model also works with ODE model
  - ➢ No obvious reason for a person doing the model to write it differently



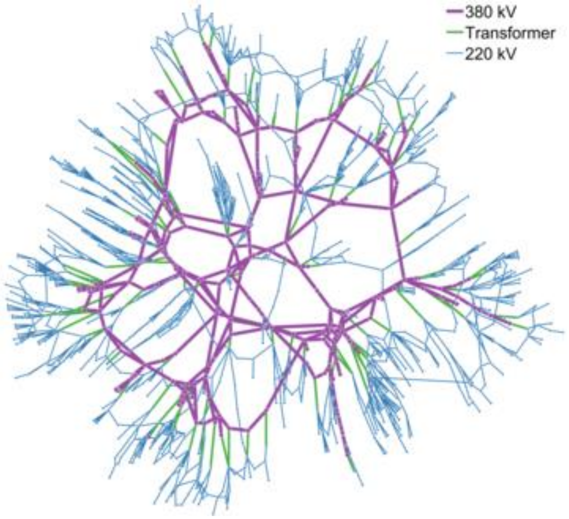
```
if running.value then
  QGenPu = if QGenRefPu >= QMaxPu then QMaxPu else if QGenRefPu <= QMinPu then QMinPu else QGenRefPu;
else
  QGenPu = 0;
end if;
```

  - ➢ Addition of a pre() could have impact on the results, depending on the tool solving strategy



```
 if running.value then
   QGenPu = if pre(qStatus) == QStatus.AbsorptionMax then QMaxPu else if pre(qStatus) == QStatus.GenerationMax then QMinPu
lse QGenRefPu;
 else
   QGenPu = 0;
```

Sparsity



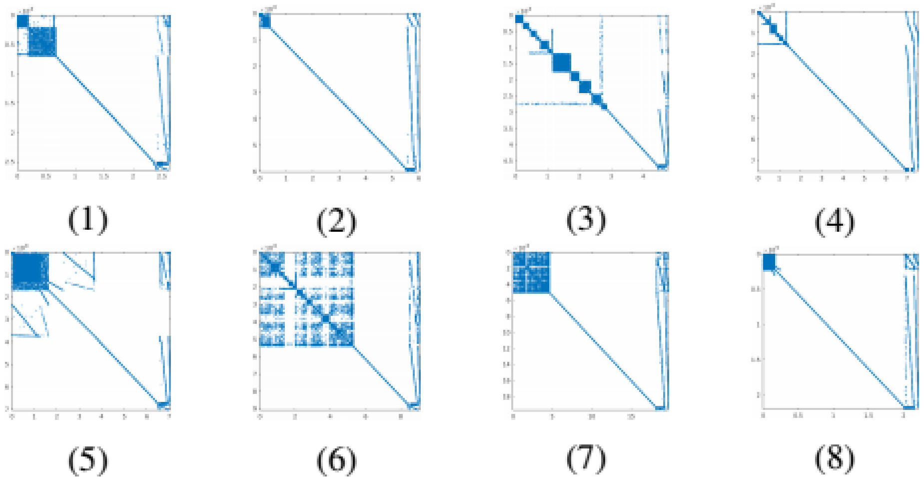


Fig. 1: Matrix sparsity patterns

- Power system – A very sparse structure by construction
  - ➢ Each bus only connected to a few other ones (a bit meshed at the transmission level, generally speaking radial at the distribution level)
  - ➢ In transient (= phasor, = electromechanical) simulation, the network part is algebraic and thus all generators see immediately any change into the system.
  - ➢ Going from DAE to ODE reduces the Jacobian size but fills it a lot.
    - ❖ Many years of work to exploit and keep the sparsity in power system and mathematical communities (Pegase European project, specific linear solver development and insertion into state-of-the-art solvers).

Sparsity

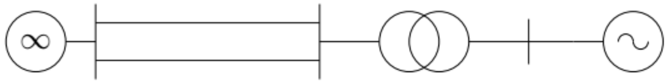


Figure 1: SMIB system representation

Mode	NNZ	Size	d (%)
ODE	32	6 * 6	89
DAE	125	32 *32	12

- Very simple test case -> Single Machine Infinite Bus test case
  - ➢ Still quite ok with ODE because all the derivatives are on the same part (generator model -> 6 states).

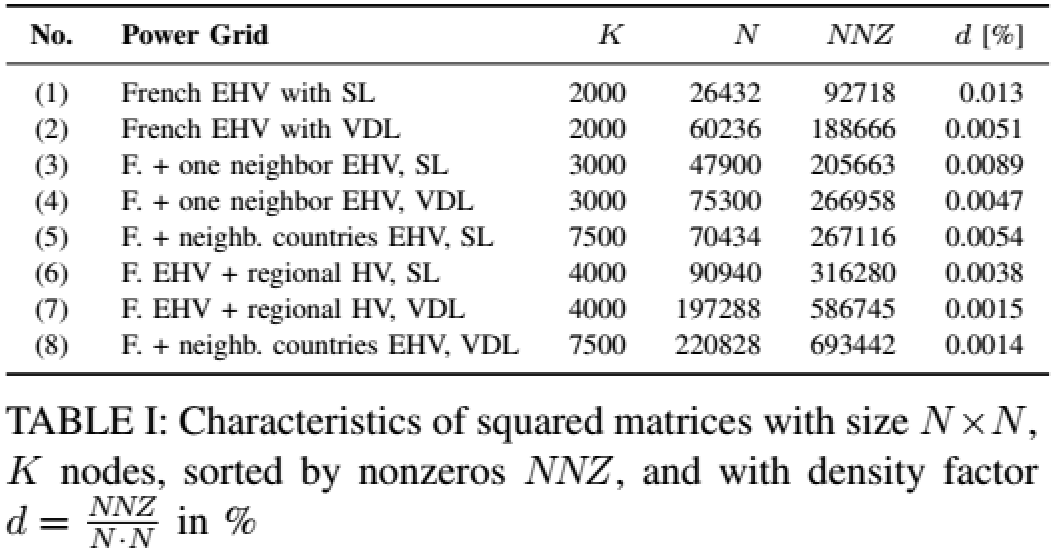
- Let’s make it a bit more complex -> Two Machines Infinite Bus test case
  - ➢ Adding another generator in parallel to the first one
    - ❖ 6 more states -> 12 states, most of them related together in ODE mode
    - ❖ The density remains similar in ODE, decreases by 2 on DAE

Mode	NNZ	Size	d (%)
ODE	122	12 * 12	84
DAE	212	54 * 54	7.2

Sparsity

- Results on larger test cases -> ScalablePowerGrids library (developed by F. Casella)

Case	*Mode*	*NNZ*	*Size*	*d (%)*
N_4_M_4	ODE	7 696	96 * 96	83
	DAE	2 138	706 * 706	0.43
N_8_M_4	ODE	30 752	192 * 192	83
	DAE	4 330	1 426 * 1 426	0.21
N_8_M_8	ODE	122 944	384 * 384	83
	DAE	8 778	2 882 * 2 882	0.10

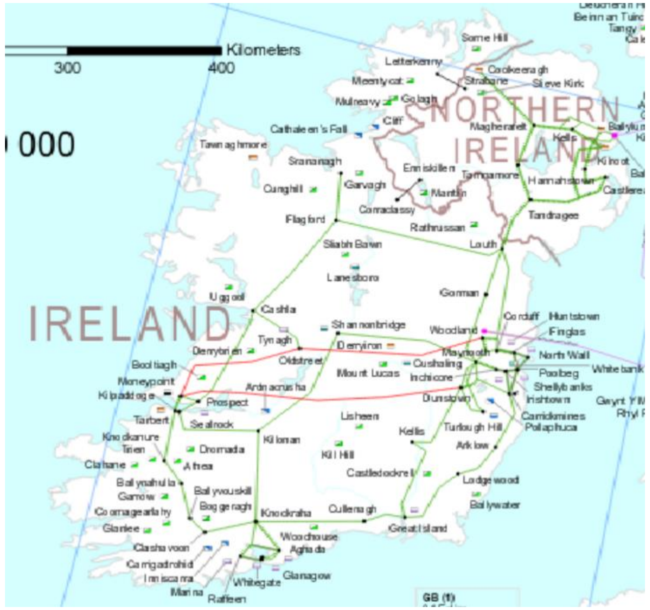


No.	Power Grid	K	N	NNZ	d [%]
(1)	French EHV with SL	2000	26432	92718	0.013
(2)	French EHV with VDL	2000	60236	188666	0.0051
(3)	F. + one neighbor EHV, SL	3000	47900	205663	0.0089
(4)	F. + one neighbor EHV, VDL	3000	75300	266958	0.0047
(5)	F. + neighb. countries EHV, SL	7500	70434	267116	0.0054
(6)	F. EHV + regional HV, SL	4000	90940	316280	0.0038
(7)	F. EHV + regional HV, VDL	4000	197288	586745	0.0015
(8)	F. + neighb. countries EHV, VDL	7500	220828	693442	0.0014

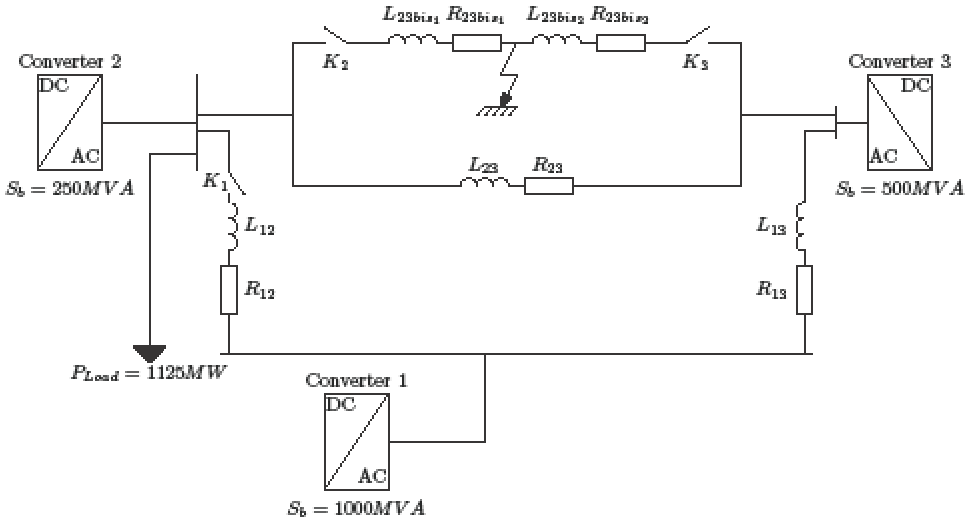
TABLE I: Characteristics of squared matrices with size $N \times N$, K nodes, sorted by nonzeros NNZ, and with density factor $d = \frac{NNZ}{N \cdot N}$ in %

L. Razik, L. Schumacher, A. Monti, A. Guironnet, and G. Bureau, “A comparative analysis of LU decomposition methods for power system simulations,” 2019 IEEE Milan PowerTech, Jun. 2019.

Performances



- Time spent in the simulation process decreases as:
  - Sparse linear solvers are working in their optimal conditions
  - « Light » causalization during compilation time and no need to go through large algebraic loops
- Results on simple and larger test cases:
  - 3 converters network with derivatives variables in the line model (EMT)
  - 3 converters network without derivatives variables in the line model (TS)
  - Ireland network



Case	*Simulation time gain* *ODE -> DAE (%)*
ThreeConv with derivatives variables	-52%
ThreeConv without derivatives variables	-54%
Ireland network	-25%

04

Conclusion

FH Bielefeld
University of
Applied Sciences

Conclusion

- Large-scale power system simulations
  - A very large and sparse network connecting components having a dynamic behavior (in TS domain).
  - Large set of hybrid, stiff and sparse index 1 semi-explicit set of equations
  - Domain-specific tools have been optimized to take advantage of this property.

  - Not a classical property for Modelica-based problems
  - Modelica-based tools not competitive with domain-specific tools
- DAE mode introduction in OpenModelica
  - Efficient implementation keeps a light causalization process for initialization and event handling
  - Stick to a DAE approach for the simulation part
  - Enables to keep the interesting natural properties of the system.

  - A mature feature - One step towards full Modelica-based tool use for large-scale power system simulations
  - Widely used into the PowerGrids library
  - Exclusively used into « Dynaωo » (RTE's industrial Modelica / C++ simulation tool) since this summer -> it works fine!
- Next steps
  - Some additional properties of power system problems could be exploited to speed-up performances (using the redundancy between components to speed up compilation for example).
  - Enrich the langage to deal with connectivity analysis during simulation

Q & A ?