Co-Simulation of Transient Effects in Superconducting Accelerator Magnets

Workshop 13-14 June 2019

Michał Maciejewski
on behalf of the STEAM team
Superconducting Accelerator Circuits – Numerical Challenges

Controller Model

- Differential-algebraic equations
- **Fixed frequency of operation**
- 10-100 elements
- Non-linear behaviour

Field model of a magnet

- Partial differential equations
- **Varying time constants**
  - ~10us (quench) - ~10ms (losses)
- **Varying geometric scales**
  - ~10um (filaments) - ~10m (magnet)
- ~10k degrees of freedom
- Highly non-linear material properties and equations

Network model of a circuit

- Partial differential equations
- **Varying time constants**
  - ~1ms (switch) - ~10min (circuit discharge)
- **Varying geometric scales**
  - ~10cm (diode) - ~10km (circuit)
- ~10k elements
- Non-linear behaviour

Busbar Model

- Partial differential equations
- **Varying time constants**
- **Adaptive mesh refinement**
- ~1k elements
- Non-linear behaviour

Multi-Domain, Multi-Physics, Multi-Rate, and Multi-Scale Problem
Some of these Multi-X phenomena can’t be simulated with the desired accuracy in a single simulation tool.

### Research Questions

1. How to represent a Multi-X problem in a consistent and generic way?
2. How to characterize the coupling between the domains?
3. What algorithm to choose in order to couple the models?
4. How to ensure consistency of the coupled simulation results?

Coupling of dedicated models promises to tackle these numerical challenges.
Presentation Outline

1. Introduction - Challenges
2. Algorithms and architecture for the cooperative simulations
3. Applications
4. Hierarchical co-simulation
5. Conclusion
Cooperative Simulation - Motivation

Suitable to study interaction of models characterized by different

- dynamical behavior: fast vs slow, fixed vs adaptive time stepping
- mathematical formulation: ordinary vs partial differential equations
- geometrical scales: phenomena occurring at micro vs macro scales, different mesh
- physical phenomena: migration between tools can be time consuming
Co-Simulation Algorithms*

One-way coupling

\[ S_1 \]
\[ x_1^{(\rho-1)}, x_1^{(\rho)}, x_1^{(\rho+1)}, \ldots \]
\[ T_{j-1}, T_j, T_{j+1} \]

\[ S_2 \]

\[ x_2(t) \]
\[ x_1(t) \]

*Courtesy prof. Sebastian Schöps, Technical University of Darmstadt
Co-Simulation Algorithms*

One-way coupling

Weak coupling

*Courtesy prof. Sebastian Schöps, Technical University of Darmstadt
Co-Simulation Algorithms*

One-way coupling

Strong coupling

Weak coupling

\[ x_1(t) \]

\[ S_1 \]

\[ x_2(t) \]

\[ S_2 \]

\[ t \]

iterate

*Courtesy prof. Sebastian Schöps, Technical University of Darmstadt
Co-Simulation Algorithms*

One-way coupling

Strong coupling

Weak coupling

Waveform Relaxation

*Courtesy prof. Sebastian Schöps, Technical University of Darmstadt
Co-Simulation Framework Architecture

**Supported tools**
- LTspice
- PSpice
- LEDET*
- COMSOL: SIGMA&BBQ
- ANSYS
- ProteCCT
- QLASA**
- PSIM

The proposed data structure and algorithm enable execution of all four co-simulation algorithms.

*Implementation of the LEDET tool adapter with the support of E. Ravaioli (LBNL/CERN)
**Implementation of QLASA tool adapter by E. Stubberud with the support of V. Marianozzi (INFN)
Tool Adapter API

Tool Adapter is a model-independent interface for simulation tools capable of:

- Setting/getting parameters
- Controlling simulation time
- Writing input
- Reading output

The use is not limited only to co-simulation!
Presentation Outline

1. Introduction - Challenges
2. Algorithms and architecture for the cooperative simulations
3. Applications
   1. Controller-Circuit Coupling
   2. Field/Circuit Coupling
   3. One-Way Coupling with Mesh-Based Interpolation
4. Hierarchical co-simulation
5. Conclusion
Controller/Circuit Coupling

1. An ideal current source is used for the studies of the LHC main dipole circuit (and others as well)
2. A first order model is used to design the power converter controller

Discrete controller is a weak coupling scheme

Waveform relaxation scheme can speed-up the computation
Field/Circuit Coupling

1. Field model is solved with FEM tool (COMSOL) to accurately resolve electromagnetic and thermal phenomena
2. Circuit model is solved with a network solver (PSpice) to represent non-linear behavior of elements
3. Field and circuit models are intrinsically coupled particularly during the magnet protection scenarios
Field/Circuit Coupling – *Waveform Relaxation*

Iterate until convergence is reached

Computational causality of lumped models

Current-driven mode

Circuit model

-field model

Voltage-driven mode

Circuit model

-field model

“Field” model has an non-preferred, differential causality

“Field” model has an integral, preferred causality
For this academic example the solution obtained with the co-simulation is in a good agreement with the monolithic one.
Field/Circuit Coupling – Building Blocks

Dedicated notebooks to create a field model, network model and a preconditioner
One-Way Coupling with Mesh-Based Interpolation

1. Magneto-thermal models are solved with COMSOL and problem-specific mesh
2. Mechanical models are solved with ANSYS and problem-specific mesh
3. Magneto-thermal load on the magnet during the energy discharge is relevant for Nb$_3$Sn superconductors
Signal Exchange from COMSOL to ANSYS
Displacement in the x direction

Displacement in the y direction

Von Mises Stress

The asymmetry of the displacement and stress due to the CLIQ discharge and initial thermal stress (hot spot).
1. Single analysis may involve a hierarchy of magnet models (lumped inductor, 1D model, 2D model)
2. Circuit operation involves several states (Controller/Circuit coupling, Field (1D, 2D)/Circuit coupling)

It is necessary to switch between coupled models and coupling algorithms (state machine)

Hierarchical Co-Simulation – HL-LHC Inner Triplet

- Co-simulation involved 8 models: 1D COMSOL, 2D COMSOL, 5x2D LEDET, 1 PSpice
- 142 signals exchanged between the coupled models

Modified LHC Main Dipole Circuit for the HL-LHC Upgrade

- We have a validated LHC Main Dipole circuit model
- M. Mentink validated an 11 T magnet with measurements:
Conclusion

1. The co-simulation framework architecture implements four main co-simulation algorithms and six tool adapters;
2. The framework was employed to develop and study dedicated co-simulation algorithms (controller/circuit coupling, field (1D, 2D)/circuit coupling)
3. Hierarchical co-simulation allows for switching of models and coupling schemes
4. Coding conventions and continuous integration ensure sustainable project development;
5. The framework has been already used for a number of analyses (LHC, HL-LHC, FCC).

You can find more information in my PhD thesis: https://cds.cern.ch/record/2675039
Getting new players – examples of LEDET/QLASA

1. Run existing model provided by the developer
   A. Run monolithically
      i. validate execution from a command line
      ii. validate I/O capabilities
   B. Run with external input over several time windows
      i. validate store/restore state
   C. Run in a coupled scheme
      i. validate information exchange with another model

2. Automatically build and run a model (optional)
   A. validate a workflow
   B. Redo 1.A-C
1. A Run existing model monolithically

Set Input – u(t)  Model  Get Output – y(t)

<table>
<thead>
<tr>
<th>Tool</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMSOL</td>
<td>I(t)*</td>
<td>R(t)<em>, U(t)</em></td>
</tr>
<tr>
<td>QLASA</td>
<td>I(t)*</td>
<td>R(t)<em>, U(t)</em></td>
</tr>
<tr>
<td>LEDET</td>
<td>I(t)*</td>
<td>R(t)<em>, U(t)</em></td>
</tr>
</tbody>
</table>

Run

Create Model object
initialiseModel()
setInput()
setSimulationStudy()
executeStudy()

getOutput()

Model (abstract class)

+ name: String
+ Model(String): Model
  + initialiseModel()
  + setInput()
  + setSimulationTime(double, double, double)
  + executeStudy()
  + getOutput()
  - saveOutput()
  - prepareInput()

*evaluated over a subset of windings (available leads)
Model description – input/output

Field Model

<table>
<thead>
<tr>
<th>l_w_name</th>
<th>U_w_name</th>
<th>R_w_name</th>
<th>Coil</th>
</tr>
</thead>
<tbody>
<tr>
<td>coil 1</td>
<td>lw 0 0</td>
<td>coil 1</td>
<td>L1</td>
</tr>
<tr>
<td>coil 1</td>
<td>lw 0 1</td>
<td>coil 1</td>
<td>L2</td>
</tr>
<tr>
<td>coil 1</td>
<td>lw 0 2</td>
<td>coil 1</td>
<td>L1</td>
</tr>
<tr>
<td>coil_1</td>
<td>lw_0_3</td>
<td>coil_1</td>
<td>L2</td>
</tr>
</tbody>
</table>

Inductors

L1

L2

Windings

Magnet 2D MQS+HT

Circuit

Can be any circuit, the name should match

\[ \begin{bmatrix} L_d \\ U_{\text{circuit}(t)} \\ U_{\text{field}(t)} \\ R_{\text{field}(t)} \end{bmatrix} \]
1.B Run with external input over several time windows

---

**Initial State** – $x_0(t)$

**Set Input** – $u(t)$

**Model**

**Run**

**Get Output** – $y(t)$

**Final State** – $x_{end}(t)$

---

*evaluated over a subset of windings (available leads)*

---

**Model (abstract class)**

- `name: String`
  - `Model(String): Model`
  - `initialiseModel()`
  - `setInput()`
  - `setSimulationTime(double, double, double)`
  - `executeStudy()`
  - `getOutput()`
  - `storeState()`
  - `restoreState()`
    - `saveOutput()`
    - `prepareInput()`

---

**Tool** | **Input** | **Output**
---|---|---
COMSOL | $I(t)^*$ | $R(t)^*, U(t)^*$
QLASA | $I(t)^*$ | $R(t)^*, U(t)^*$
LEDET | $I(t)^*$ | $R(t)^*, U(t)^*$

---

**Create Model object**

- `initialiseModel()`
- `setInput()`
- `setSimulationStudy()`
- `executeStudy()`

---

**Store State**

- `storeState()`
- `getOutput()`

---

**Time**
1.C Run in a coupled scheme

Application: Field-Circuit Coupling – Gauss-Seidel Method - Field (MQS+HT)

Waveform relaxation (Gauss-Seidel Method):

Block coil v20ar

More details in M. Prioli, EuroCirCol WP5 Workshop 2016 link
1.C Run in a coupled scheme

Application: 1D Quench Initiation and Propagation

\[ t_k = t_{k-1} + \Delta t_{PID} \]