



HAL
open science

Contributions théoriques à la co-simulation EMT-TS

Helena Shourick

► **To cite this version:**

Helena Shourick. Contributions théoriques à la co-simulation EMT-TS. Mathématiques générales [math.GM]. Université Claude Bernard - Lyon I, 2023. Français. NNT: 2023LYO10002. tel-04390142

HAL Id: tel-04390142

<https://theses.hal.science/tel-04390142v1>

Submitted on 12 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



THESE de DOCTORAT DE L'UNIVERSITÉ CLAUDE BERNARD LYON 1

Ecole Doctorale N° 512
Informatique et Mathématiques de Lyon

Spécialité de doctorat :
Mathématiques Appliquées

Soutenue publiquement le 05/01/2023, par :
Hélène Shourick

Theoretical contributions to EMT-TS co-simulation

Contributions théoriques à la co-simulation EMT-TS

Devant le jury composé de:

Dolean, Victorita	Maître de conférence	Université Côte d'Azur	Rapporteure
Mahseredjian, Jean	Professeur	Polytechnique Montreal	Rapporteur
Debit, Naïma	Maître de Conférence	Université de Lyon 1	Examinatrice
Magoules, Frederic	Professeur	CentraleSupélec	Président du jury
Tromeur-Dervout, Damien	Professeur	Université de Lyon 1	Directeur de thèse
Chédot, Laurent	Docteur	SuperGrid Institute	Invité

Remerciements

Bien que ce manuscrit soit signé de mon nom, il résulte du travail et du soutien d'un grand nombre de personnes. Je saisis l'occasion de les remercier ici, en commençant par mon directeur de thèse Damien Tromeur-Dervout qui m'a apporté une aide considérable tout au long de ces 3 années, qui s'est montré extrêmement disponible et bienveillant. Je souhaiterais aussi remercier mon encadrant industriel Laurent Chédot pour son suivi régulier, son soutien et ses conseils avisés qui m'ont permis de prendre confiance en mes compétences et mon travail.

Je souhaiterais exprimer ma reconnaissance aux membres du jury pour s'être rendus disponibles afin d'évaluer mon travail, Maître de conférence Victorita Dolean et Professeur Jean Mahseredjian pour avoir relu mon rapport, Maître de conférence Naïma Debit pour avoir examiné mon travail et Professeur Frédéric Magoules pour m'avoir fait l'honneur de présider le jury. Je souhaite en particulier les remercier pour leur clairvoyance lors de la soutenance.

Je voudrais aussi remercier mes collègues de Supergrid Institute pour les discussions à l'espace café et leur bonne humeur. Merci en particulier aux copains du programme Architecture et systèmes pour nos jeux, nos fous rires, vous m'avez donné envie de venir au travail même lors des périodes de démotivations, j'ai adoré vous rencontrer, j'ai adoré votre énergie, vous êtes tous super.

Enfin, je souhaiterais remercier ma famille et mes proches, sans qui je ne serai simplement pas arrivée à ce niveau d'étude. Vos visages m'ont manqué pendant ces périodes de confinement, malgré tout, vous avez rendu cette période de covid moins amère. Merci d'être vous, merci pour vos soutiens, vos sourires, votre amour, merci de me remettre les idées en place au besoin.

Après vous avoir remercié pour votre soutien, je me rends compte que finalement, il serait plus juste de remercier cette thèse de m'avoir permis de faire des rencontres extraordinaires et m'avoir fait réaliser à quel point j'étais soutenue. Vous m'avez fait grandir. Merci du fond du cœur.

Contents

1	Introduction	1
1.1	General context	1
1.2	Transients phenomena in electrical networks	3
1.3	Mathematical modeling of electrical networks	5
1.4	Optimization and acceleration efforts	8
1.4.1	Parallelization of computations	8
1.4.2	Hybridization of EMT models and low level of detail models	9
1.5	State of the art on EMT-TS co-simulation	10
1.5.1	The partitioning	10
1.5.1.1	Based on line	11
1.5.1.2	Coherency based partitioning	12
1.5.1.3	Graph based partitioning	14
1.5.2	The interaction protocol	15
1.5.3	Exchanging Values	16
1.5.3.1	Transformation of the values	17
1.5.3.2	Dynamic equivalent	17
1.6	Outline of the thesis & Contributions	19
2	Schwarz method in the homogeneous EMT or TS case	21
2.1	Introduction	22
2.2	The Schwarz domain decomposition method	24
2.2.1	Generalized Schwarz alternating method	24
2.2.2	Connection between the Steklov-Poincaré operators and the Schur complement	26
2.3	Schwarz method for electrical circuit DAE	27
2.3.1	Time discretizing of linear electrical circuit	28
2.3.2	RAS splitting in Detail	29
2.3.3	RAS algorithm	29
2.4	Error operator and Acceleration of convergence	30
2.4.1	The RAS error operator	31
2.4.2	The Aitken's acceleration of convergence of the RAS	32

2.4.3	Acceleration of the RAS in the electrical circuit context . . .	34
2.5	Numerical results	36
2.6	Conclusion	42
3	Heterogeneous EMT-TS RAS	45
3.1	Introduction	45
3.2	Schwarz method for heterogeneous EMT-TS	47
3.2.1	TS side	47
3.2.1.1	Translation from EMT to Dynamic Phasor	49
3.2.2	EMT side	52
3.2.2.1	Translation from Dynamic Phasor to EMT	54
3.2.3	Initialization	56
3.2.4	Heterogeneous EMT-TS RAS formulation	57
3.3	Heterogeneous EMT-TS RAS error operator	57
3.3.1	Acceleration strategy	58
3.3.1.1	Numerical Computation of the error Operator	58
3.3.1.2	Impact of heterogeneity in the choice of acceleration strategy	59
3.3.2	Error: a multifactorial resultant	59
3.3.2.1	Impact of the translation operator from EMT to TS	60
3.3.2.2	Impact of the translation operator from TS to EMT	60
3.4	Heterogeneous EMT-TS Numerical Results	61
3.4.1	Heterogeneous EMT-TS RAS convergence results	61
3.4.1.1	Effect of circuit topology on the convergence	61
3.4.1.2	Effect of the EMT Time steps Δt_{emt} on the convergence	64
3.4.1.3	Effect of the translation operators on the convergence	67
3.4.1.4	Effect of the reduction of Δt_s and moving history on the convergence	69
3.4.1.5	Largest eigenvalue of the heterogeneous EMT-TS RAS error operator	71
3.4.1.6	Aitken's acceleration of the heterogeneous EMT-TS RAS convergence	72
3.4.2	Qualitative results on the heterogeneous EMT-TS RAS	75
3.4.2.1	The advantage of the EMT part	75
3.4.2.2	The impact of the cutting and passing of information	76
3.5	Conclusions	77

4	Dynamic Iteration	79
4.1	Introduction	80
4.1.1	Motivations	80
4.1.2	Previous works for accelerating Dynamic iteration	80
4.2	Dynamic Iteration	81
4.3	Dynamic Iteration with RAS splitting	86
4.3.1	DI with RAS splitting in the continuous case	86
4.3.2	Discrete counterpart of the DI with RAS splitting	89
4.4	Time stepping strategies for DI	90
4.4.1	Sequential DI strategy	90
4.4.2	Pipelined DI strategy	91
4.4.3	Link between the error operators of sequential DI and pipelined DI	93
4.4.4	Comparison between Sequential and pipelined strategy	95
4.5	Numerical Results on DI	98
4.5.1	Sequential strategy	98
4.5.2	Pipelined DI with RAS splitting strategy	100
4.6	Conclusion	103
5	Architecture	107
5.1	Introduction	107
5.2	OpenModelica	108
5.2.1	Modelica language	109
5.2.2	OpenModelica Advantages	110
5.2.3	OpenModelica disadvantages	113
5.2.4	Conclusion about OpenModelica	114
5.3	Functional Mockup Interface	115
5.3.1	OpenModelica FMU's export	116
5.3.1.1	Interface components	116
5.3.1.2	Causality of interface data	120
5.3.1.3	Other problems encountered	122
5.3.1.4	Conclusions about FMI	124
5.4	Orchestrator	125
5.4.1	General Structure	126
5.4.1.1	Communication protocol for the platform	127
5.4.2	FMU type slave	130
5.4.2.1	FMU slave Management	133
5.4.2.2	C++ wrapping test	134
5.4.3	Conclusion About the Orchestrator	136
6	Conclusions and perspectives	137

Bibliography	141
List of Figures	153
Résumé en Français	159
Abstract	163

Chapter 1

Introduction

Contents

1.1	General context	1
1.2	Transients phenomena in electrical networks	3
1.3	Mathematical modeling of electrical networks	5
1.4	Optimization and acceleration efforts	8
1.4.1	Parallelization of computations	8
1.4.2	Hybridization of EMT models and low level of detail models	9
1.5	State of the art on EMT-TS co-simulation	10
1.5.1	The partitioning	10
1.5.2	The interaction protocol	15
1.5.3	Exchanging Values	16
1.6	Outline of the thesis & Contributions	19

1.1 General context

At a time when ecology is at the heart of concerns, the integration of renewable energies into the network is increasing. Moreover, the share of these energies should increase further, at least in Europe, over the next decade in order to reach the European Union's objective of 45% renewable energies in its energy consumption by 2030 (revision of the Renewable Energy Directive (RED) voted on September 14, 2022 by the European Parliament).

In addition, networks are increasingly interconnected. In Europe, for example, it was recently voted that each member country must have at least two cross-border projects. Thus, High Voltage Direct Current (HVDC) lines are created to connect unsynchronized electrical systems, or electrical systems connected over very long distances, Celtic Interconnector, linking France and Ireland is an example of this type of project in progress. HVDC lines are also being created to transmit power generated in new renewable energy resource parks located in the least populated areas (example: Sahara wind project). Power systems are mainly Alternating current (AC), the HVDC lines are connected to the networks through power electronic converters.

This development of renewable energies and HVDC links therefore leads to the presence of more and more power electronics components in the networks. This type of component implies faster transients than those implied by the historical infrastructure. However, in a world where most things run on electricity, the security and reliability of the power grid is a priority. Proper modeling and simulation of these systems is essential to ensure their safety and sustainability. Indeed the simulation allows to predict the situations which can occur in the future without spending a lot of money in non-virtual tests. This also makes it possible to check the correct calibration of the components and thus to have more durable networks over time.

SuperGrid Institute

This PhD was carried out within the SuperGrid Institute. SuperGrid Institute is a French institute for the energy transition (ITE). This independent research and innovation center combines industrial expertise and public research and relies on private and public shareholders.

The research and innovation work carried out within the SuperGrid Institute is dedicated to the development of the electricity transmission network of the future, indeed SuperGrid Institute intends to be an important player in the transformations of the network aimed at including more renewable energy resources.

The production of renewable energies being, in most cases, located far from the places of consumption, the development and improvement of High Voltage Direct Current and Medium Voltage Direct Current lines allowing the transport of a large quantity of energies over long distances is an important issue in the transformation of networks. This is why research on this type of technology is a major activity at SuperGrid Institute.

Thanks to the presence of different expertise profiles, SuperGrid presents a multidisciplinary approach offering a wide range of services and solutions for the development of electrical systems, equipment and components. SuperGrid Institute's test platforms also allow it to offer large-scale studies

and simulations.

SuperGrid Institute is made up of several research programs:

- Supergrid Architecture & Systems, in this program, the focus is on high and medium voltage direct current control and protection strategies.
- High Voltage Substation Equipment, in this program substation technologies that meet the constraints of future DC networks and those of current AC networks are developed.
- Power Electronics & Converters, in this program, power transmission solutions networks are developed, research in this program also covers AC/DC and DC/DC converters.
- HVDC Cable systems & Junctions, in this program, high-performance insulation materials meeting the constraints of HVDC are developed.
- Power Storage & Balancing, in this program, hydraulic storage technologies are adapted and developed to support the integration of renewable energies within the European electricity network.

This PhD was carried out within the Architecture and Systems program, in fact, research on the improvement of the simulation of electrical systems, necessary for research in the field of power transmission systems is one of the activities of this program. More precisely, this thesis took place in the Modeling and Simulation team, whose objective is to develop models and simulation platforms for the study of HVDC networks.

The expectations of the SuperGrid Institute regarding this thesis were first of all to co-simulate the EMT and TS models in order to improve the simulation of networks but also to bring a mathematical point of view on this subject. Another objective was to develop skills on the Modelica language and to get a deeper insight on the OpenModelica tool suite.

The following section will present the main categories of network transient and their modelisation.

1.2 Transients phenomena in electrical networks

As introduced previously, the different types of equipment present on a network involve more or less rapid transients. There are several main categories of transients in power systems [72]:

- **Electromechanical transient:** predominant transient in the network, the system is assumed to remain in quasi-steady state. This type of transient is simulated using a Transient Stability (TS) type modelisation. This type of simulation can simulate low frequency disturbances and be used to simulate millisecond scale events. Based on the observation that the fast transients attenuate and then quickly disappear, the Phasor type simulation simply ignores them. Transient stability simulation captures dynamics down to 5 Hz. PSS/E software allows simulating networks with this type of simulation.
- **Electromagnetic transient:** these are the fastest transients (of the order of a microsecond) and which require the finest modeling: modeling of the electromagnetic transient type (EMT). This type of modeling consists in keeping the complete DAE system and solving it. This makes it possible to capture all the transients and to obtain the most detailed simulation possible of the electrical network and to monitor all of its dynamics. an EMT time step size is a few tens of microseconds (unless there is faster dynamics that requires smaller time steps). Emt-type simulation is generally used to simulate small networks in which fast dynamics occur because the big disadvantage of this type of simulation is that it requires a lot of resources and more calculation time. Examples of software allowing simulation with this type of modeling are: EMTP, PSCAD, RTDS and Hypersim.

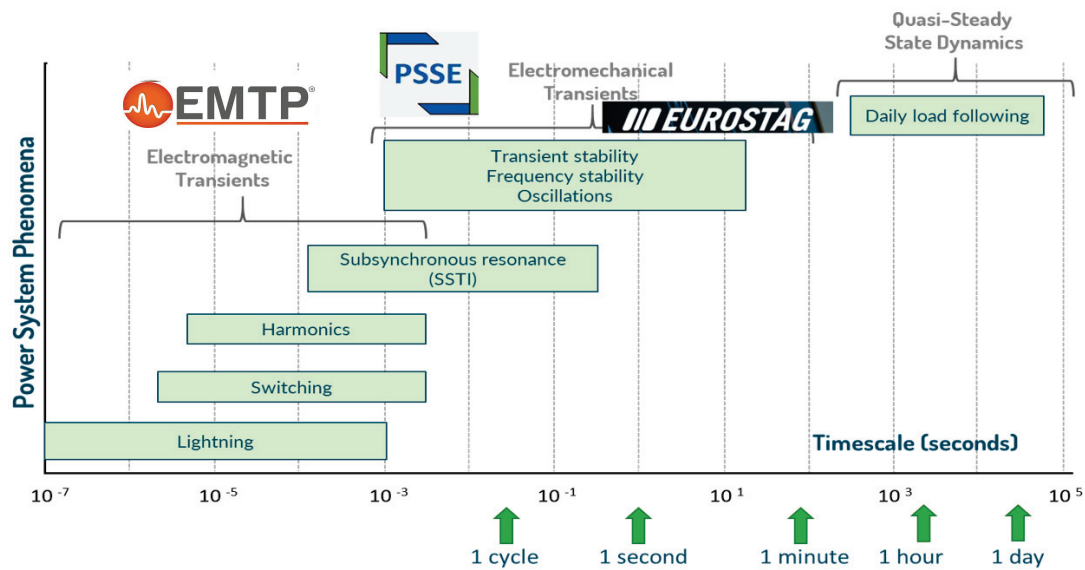


Figure 1.1: Time frame of power system transients [93]

We also compute the operating point in **Steady-state** by methods of the load-

flow type. These methods are generally used to simulate balanced transmission systems. They are useful for the initialization of simulations in order to start from equilibrium conditions. Figure 1.1 shows how power system phenomena are studied based on the time scale at which they occur.

For large electrical systems, EMT simulation appears to be limited in terms of simulation time because very small time steps must be used to capture highly oscillatory phenomena. For example, the comparison between fixed and variable time steps solvers on an IEEE118 test case with 118 Buses in the article by Masoom & al for highly oscillatory phenomena shows that this type of simulation is quite expensive when the required time steps are very small [75]. Commercial and non-commercial tools exist to model power systems. Some tools allow you to create components from the equations that govern its behavior and then connect these components together to create a network. The tool then takes care of creating the DAE system corresponding to the network. For this thesis, the choice fell on the OpenModelica suite of tools. This choice was made for the transparency of these tools, the fact that it is based on components and that they use the Modelica language which is increasingly used in the electrical industry and finally this tool is open source. In this way, investigating what is possible to do with this tool is an integral part of the objectives of this thesis.

1.3 Mathematical modeling of electrical networks

The electrical network can be viewed as a graph connecting electrical components through their connecting pins. The vertices or nodes of this graph are the pins of electrical components and its edges are the link between vertices of two connected components. Some physical quantities are defined on this pins such as currents and voltages. The common principle of different mathematical modeling is based on the application of Kirshoff's laws, which establish the mathematical relationships between the different physical quantities of an electrical network. Kirshoff's laws are as follows:

- Kirshhoff's current law: The sum of the currents entering a node is equal to the sum of the currents leaving this node.
- Kirshhoff's voltage law: The sum of the voltages around any closed loop is zero.

The nodal analysis express the potential at each node using Kirshkoff's laws and the component properties of the branches connected to that node. It creates

an admittance matrix linking the sum of the current entering each node and the voltages. However, the nodal formulation does not allow to directly represent certain devices as current-dependent circuit element [53].

The Modified Nodal Analysis, introduced by [53], widely used in network modeling since [114], then the Modified Augmented Nodal Analysis presented in [72], improved especially for modeling electromagnetic transients [71], allows one to overcome those difficulties. This way the complete network is written in the form

$$A_{N_t} z_{N_t} = b_{N_t},$$

where A_{N_t} is the linearized matrix at time t , if there are nonlinear devices, the admittance matrix of the nodal analysis is included in the matrix A_{N_t} . The z_{N_t} are the current and voltages unknowns and b_{N_t} contains the knowns current and voltages at time t . Notes that in this formulation the time discretization has already been performed.

Another method often used to obtain the system representing the dynamics of the network is the state space analysis, it is a method widely used in physics. The general form of state space system is :

$$\begin{aligned} \dot{v}(t) &= \mathbf{A}v + \mathbf{B}u \\ w &= \mathbf{C}v + \mathbf{D}u \end{aligned} \quad (1.1)$$

where v are the state variables, u the inputs and w the outputs. The state matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ are linear matrices or some linearizing around a time t of the components behavior in the electrical network and must be recomputed if nonlinear components are involved or if there are topology changes in the network.

The more general mathematical formulation of the electrical network considers the construction of the differential algebraic equations system induced by the Kirshoff's laws and the electrical network's components modeled using differential equations:

$$F(t, x(t), \dot{x}(t), y(t)) = 0 \quad (1.2)$$

where $x(t) \in \mathbb{R}^n$ are the differential unknowns, $\dot{x}(t) \in \mathbb{R}^n$ are the derivatives of x with respect to time and $y(t) \in \mathbb{R}^m$ are the algebraic unknowns. Some tools based on Modelica language generate such DAE system.

To find a form of DAE system more similar to the state space system, one can separate in the DAE system the purely algebraic equations from the others. This way the general DAE system (1.2) can be rewritten as:

$$\begin{cases} \dot{x}(t) = f(t, x(t), y(t)) \\ 0 = g(t, x(t), y(t)) \end{cases} \quad (1.3)$$

The mathematical modeling of the electrical network depends also on the nature of transient phenomena that must be caught. Certain assumptions made or not

made about the shape of the unknowns can lead to different levels of mathematical modeling.

If no hypothesis is made on the shape of unknowns, then the DAE system formulated by system (1.2) will be solved to address some EMT stability. The presence of fast dynamics strongly constrains the time step length of the DAE numerical solver.

A strong hypothesis on the shape of the unknowns can simplify the system (1.2). The phasor modeling considers the electrical unknowns as sinusoidal signal with a given constant frequency ω_0 and a phase θ and can be represented by a complex constant amplitude A . The time derivative is then applied to the sinusoidal part of the phase vector. It results on a simplification of system (1.2) in a linear system. If steady-state solutions are searched, the time derivatives in system (1.2) can be omitted.

Lately another type of simulation which is a compromise between phasor simulation and EMT simulation has been developed: simulation using dynamic phasors. A dynamic phasor is a phasor whose magnitude $A(t)$ and phase angle $\theta(t)$ are time-dependent values. Dynamic phasors for electric network simulation construction is presented below, using the notations of [16], [61]:

Due to the Fourier transform, a periodic waveform $x(\tau)$ can be written on the interval $\tau \in [t - T, t]$ where T is the observation period considered, as

$$x(\tau) = \sum_{k=-\infty}^{+\infty} X_k e^{jk\omega_0\tau}$$

. Now consider that our waveform is not strictly periodic (in an almost periodic state), so the Fourier coefficients would be time-varying:

$$X_k(t) = \frac{1}{T} \int_{t-T}^t x(\tau) e^{-jk\omega_0\tau} d\tau = \langle x \rangle_k.$$

We have this following properties:

- $\langle \frac{dx}{dt} \rangle_k = \frac{d\langle x \rangle_k}{dt} + jk\omega_0 \langle x \rangle_k$
- $\langle xy \rangle_k = \sum_i \langle x \rangle_{k-i} \langle y \rangle_i$

In power electric it's this Fourier coefficient $X_k(t)$ which is conventionally called Dynamic phasor. The DAE system (1.2) is transformed into another DAE system with putting each unknown as a sum of dynamic phasors. The sum range depends on the harmonic kept. As harmonics are computed separately there is a multiplication of variables. Despite this enlargement of the DAE system, as time varying Fourier coefficients are slower than the original values, it allows time step much bigger than EMT (2 to 30 times larger). Demiray [28] discusses on the feasibility

of using Dynamic phasor to simulate large networks, dynamic phasor simulations type allow to catch dynamics up to 60Hz with a computation time reasonable for large networks. Hassani & al [51] tested dynamic phasors and the results they obtained are that this type of simulation has no advantage over a simulation in the time domain.

Table 1.1 is a summary of the modeling of the three previously presented simulation type for the basic components.

	EMT	dynamic Phasor	Phasor
system	$F(t, x(t), \dot{x}(t), y(t)) = 0$	$\langle F(t, \langle x \rangle(t), \langle \dot{x} \rangle(t), \langle y \rangle(t)) = 0$	$\tilde{F}(t, \mathbf{X}(t), \mathbf{Y}(t)) = 0$
variable	free shape	$\langle x \rangle(t) = \sum_{k=0}^m \bar{x}_k(t) e^{j\omega_k t + \phi_k(t)}$	$\mathbf{X}(t) = \sum_{k=0}^m \bar{\mathbf{X}}_k e^{j\omega_k t + \phi_k}$
resistance	$u = Ri$	$\tilde{u}_k = R\tilde{i}_k$	$\tilde{u} = R\tilde{i}$
inductor	$u = L \frac{di}{dt}$	$\langle u \rangle_k = L \langle i \rangle_k + L \langle i \rangle_k k j \omega_o$	$\tilde{u} = L \tilde{i} j \omega_o$
capacitor	$i = C \frac{du}{dt}$	$\langle i \rangle_k = C \langle u \rangle_k + C \langle u \rangle_k k j \omega_o$	$\tilde{i} = C \tilde{u} j \omega_o$

Table 1.1: EMT, phasor and Dynamic phasor representation

1.4 Electrical network simulation optimization and acceleration efforts

Several axes have been explored to speed up the simulation and/or bring more precision:

1.4.1 Parallelization of computations

A lot of work has been done to propose the possibility of executing several parts of the electrical network simulation in parallel. Platforms have been developed, based on the Functional Mock Up (FMI) interface standard [21], like Daccosim [34] which is a platform on which a lot of effort is made for the sharing of tasks, or like the MasterSim platform [89] which is based on an iterative process and which thus increases the stability of the simulation coupled, or like the InSystemLab (ISL) platform which allows running several co-simulation sessions in parallel.

Some others are using Domain Decomposition Method to perform homogeneous Co-simulation. Domain decomposition methods are methods where a problem is divided into several sub-problems and the resolution of the global problem is done in two stages, the first being the local and independent resolution of each sub-problem and the second step being the resolution boundary problems between each "neighbor" subproblem. Aristidou & Al [4, 9, 10, 7] developed two parallel algorithms based on the Schur complement. He integrated these methods into the

platform RAMSES [5, 6, 8] which makes it possible to launch a Phasor-Phasor type co-simulation in parallel in an efficient manner.

Waveform relaxation type methods are also used for solving electrical circuits since Lelarasme did [65]. Waveform relaxation methods are iterative methods consisting in cutting a system into subsystems, and, starting from an initial estimate of the solution over the complete time interval, an approximation is obtained at each iteration closer and closer to the real solution.

In each subdomain, the coupling terms are given in the form of signals obtained from neighboring subdomains at the previous iteration. Magoules & al studied the use of this type of method on large power systems [98], they concluded that the convergence of the method is particularly slow, especially if there are many subsystems. They therefore proposed an initialization method and a preconditioning method which make it possible to drastically reduce the number of iterations necessary. In his PhD Khumbar studied the use of Optimized Waveform Relaxation (OWR) methods on electrical systems, and worked on the back-substitution method, a method based on Schur's complement, which allows to reduce large circuits to smaller ones [41]. A more in-depth state of the art on these methods and their use in electrical systems has been carried out in chapter 4.

1.4.2 Hybridization of EMT models and low level of detail models

To add detail in the simulation without adding too much computational weight, EMT-TS hybridization was explored. For example, Cossart [26] deletes the differential part of the corresponding equations at the place where he considers not to need a high level of detail. E-TRAN [27] is a tool that was developed to "translate" a TS model into a dynamic equivalent for inclusion in an EMT simulation. Therefore, the objective of these hybridizations is to take advantage of both the level of detail of the EMT and the computational efficiency of the less detailed model. With this type of hybridization, the system is not split and the TS and EMT parts are solved together in a global problem. We speak of co-simulation when the problems are solved separately, then interfaced.

EMT-TS hybrid co-simulation was also studied, and the difficulties were noted by Jalili-Marandi& al [55]:

- The partitioning: with a good detail/efficiency compromise, in fact the EMT system, located in areas requiring a high level of detail, must be quite small. The cutting must be far enough from potential disturbances both to capture possible disturbances with the EMT modeling and for the stability of the method, indeed if the interface is too close to the disturbance, this can cause

problems in the data transmission. The decomposition must also allow the method to converge.

- The interaction protocol: the protocol can be serial or parallel. The updates of the sub-models with the data retrieved from the other sub-model which can be synchronized or not and more or less frequent. The protocol can be iterative or not. The choice of exchange times must take into account the difficulty of the fact that the two types of models do not use the same time steps.
- The translation from one type of model to another: indeed, the TS and EMT variables not having the same form, they must be transformed for the exchange of information.

These transformations should not be too computationally expensive and as accurate as possible. In some cases, we cannot pass only discrete values, so a full or partial representation of the other submodel is created in each sub-model. These representations are in the form of dynamic equivalents. They are especially useful for those using component-based tools where discrete values, on its own, cannot be passed.

The main objective of this thesis being the EMT-TS co-simulation, the following section will then be a state of the art dealing with the various locks related to the co-simulation mentioned above.

1.5 State of the art on EMT-TS co-simulation

1.5.1 The partitioning

One of the first steps to perform EMT-TS co-simulation is network partitioning. The location of the interface has an impact on accuracy and performance. Indeed, the EMT type simulation requires much heavier calculations than the TS simulation, it is therefore necessary to try to minimize the part of the network simulated in EMT, but also to try to take into account the principles of parallel calculation to take advantage of the architecture of the machine. To guarantee accuracy, it is necessary to identify precisely the parts to be simulated in detail but also to pay attention to the location of the interface in relation to the location of the disturbance. The first idea in the literature was to minimize the EMT part as much as possible, as in [52] which modeled an HVDC link in an AC system and which naturally located the interface buses to the terminal converters. This idea has in fact the big advantage not only of minimizing as much as possible the part simulated in EMT but also the number of interface buses. On the other hand

this idea does not take into account the phenomena of harmonic distortion and the asymmetric defects which when are located near the interface generates imbalances. To overcome this problem, it was proposed to widen the area calculated in EMT, the challenge being to know how far it is optimal to extend the detailed area so that the hybridization remains advantageous in terms of cost of calculation and that imbalance phenomena disappears [55].

1.5.1.1 Based on line

A common way to split the power grid is to cut transmission lines. One can obtain a rapid and quite efficient network partitioning based on the natural delay induced by the transmission lines and the network connectivity (the subnetworks must have as few interactions as possible between themselves) [36]. when the network is cut at transmission lines, then, thanks to the natural delay the current and voltage informations to exchange from one subsystem to another, will only count as "history current" in the nodal equations of the receiving subsystem [82].

As the delay allows the information to be consider as "history current", the delay must necessary be at least as long as a time step. In [82] the crossing time of the transmission line must be equal to the time step of TS subsystem (which must also be a multiple of the EMT solver's time step); Thus, time-steps of the two solvers are necessary time-invariant, but, in addition it is chosen taking the length of the transmission line into account rather than dynamics present in the networks.

Le-Huy & al [63] go even deeper in exploiting transmission lines properties: they still used transmission lines to partition but also as real "interfaces" between the subsystems and the two (EMT and TS) types of simulation. Indeed, real transmission lines are modelise as "hybrid lines" which will contain the representation translator.

The two most positive points in this modelisation, is firstly that its hybrid line allows informations to directly pass from EMT subsystem to TS subsystem, without the need of equivalent systems, and secondly that the location of the fault has no impact on the location of the interface, indeed in generally the fault must happens far enough from the interface [55]. One of the downsides being that as the natural decoupling of the wave which occurs in transmission lines is used to separate the harmonics in order to choose the one to keep for the phasor representation, the TS subsystem can only be on the receiving side of the line. It adds another geographic constraint for partitioning. The partitioning according to the transmission lines takes advantage of a physical phenomenon which will be present and need to be taken into account in power system modeling. Despite this, it has the disadvantage of posing a geographical strong constraint. This constraint is not so inconvenient when the subsystem are simulated with the same simulation type [36]. But as soon as this constraint is not based on the dynamics distribution in

the networks, if the aim is to simulate subsystem with different simulation type explicitly to catch different transients, it's becoming quite uncomfortable. There is as well a strong correlation between the length of the transmission line interfacing two subsystems and the time steps used by the two solvers [82, 63].

1.5.1.2 Coherency based partitioning

Some methods that have been developed for network partitioning consist of identifying and separating the oscillation modes of dynamics, these methods are based on modal analysis and coherency methods [3]. These two methods types were first designed to build low frequency equivalents. They are based on the analysis of eigenvalues of the linearized state space equation of a network:

$$\begin{aligned}\dot{v}_l(t) &= \mathbf{A}_l v_l + \mathbf{B}_l u_l \\ w_l &= \mathbf{C}_l v_l + \mathbf{D}_l u_l\end{aligned}\tag{1.4}$$

The eigenvalues of A_l furthest from the origin are related to the most damped quantities, the events in the parts of the network represented by these equations therefore disappear more quickly. In the modal methods, the less damped modes are extracted. Coherency methods consist of keeping only the swing equations, then reconstructing a state space system with these equations. Identifying in this system the degree of dependence between the different eigenvalues makes it possible to identify and group the machines into coherent groups (to construct a low-frequency equivalent, replace each group by a large equivalent machine). This method makes it possible to group together the strongly coupled machines in the same subsystem and that way, to cut on the weakest links. These two types of methods are also called two-time scales methods because they are based on the separation of the system dynamics into slow dynamics and fast dynamics.

Many partitioning methods are inspired by coherency or modal methods. Znidi & al [116] developed a coherency bus method by considering that the dynamic response of a generator after a disturbance directly influences the variation of the phase angle of neighboring buses. A PPCC (Pearson Product -moment Correlation Coefficient) is implemented to quantify the degree of consistency between each pair of buses. The problem is then transposed into a graph partitioning problem by placing the buses as vertices and the PPCC between two buses as the weight of the connection between these two buses (vertices). The graph is partitioned using HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [24]. Raak & al [100] also partitions the network using bus coherency based on voltage angle dynamics, the main difference is that the calculation is based on koopman modes and can therefore be performed on a non-linear system. The major drawback of these methods is that to calculate the tension angle dynamics either the network has been previously fully simulated or all the data has already

been measured on the real network (and therefore the disturbance has already produced on the real network).

The Modal method was used by Cossart in his PhD [26] to identify the state variables which participate the most in each pole (eigenvalue of A_l). The state variables will be grouped by group of poles in which they participate. The state variables participating in poles that one has chosen to neglect will be residualized, that is to say that a diagonal matrix \mathbf{E} called residualization matrix is built and is going to be applied to \dot{v} . $\mathbf{E}(i, i) = 1$ if the differential part of the equations associated with the variable $v(i)$ is to be kept, if $\mathbf{E}(i, i) = 0$, $v(i)$ is residualized (the differential part of the equation is neglected, $v(i)$ is no longer differential but algebraic variables), which amounts to applying the approximation of phasor to the poles to be neglected. Choosing the pole to preserve is choosing the dynamic to preserve, and for this Cossart developed three strategies.

The first strategy consists in simulating with details only the least depreciated modes. Therefore neglect the variables associated with the poles whose real parts are the farthest from the origin. This method has the advantage of being easy to implement, fast (once all the eigenvalues are computed) and direct. On the other hand, this strategy proposes to neglect the fastest dynamics, which can also cause damage on the networks. In addition, these fast dynamics, unlike the slower ones, cannot be catch with a Phasor simulation but can be with an EMT type simulation. The second strategy consists in calculating, using Hankel Singular Values (HSV), the most reachable states in the balanced realization, once these states identified, it remains to identify which poles are linked to these states using participation factors and finally to group the state variables participating in these poles (residualizing the others). Therefore, it is necessary to calculate beforehand the HSV and the balanced realization in order to identify the most observable states, the calculation of these elements is not very heavy in computing time, for that reason, the method is quite straightforward. On the other hand if the system is not very decoupled the model will not be very reduced (the EMT part can be too large). To calculate the participation factors, it is not necessary to linearize the network, indeed, Netto & al [87] set up the calculation of the participation factors using the koopman modes.

Finally, the third strategy consists in minimizing an error criterion. Error criterion obtained by applying the Fourier transform to the linearized system of equations and to the system of equations to which a residualization matrix (a diagonal matrix whose coefficients are 0 if the associated state variable is to be residualized and 1 otherwise) is applied. The main disadvantage of these method is that to apply these strategies, all the eigenvalues of the system must be obtained before the simulation, whereas the calculation of all eigenvalues of a system can be very heavy for large networks.

Partitioning can also be linked to the method used, this is the case of Tsuji & al who developed a partitioning method base on the analysis of eigenvalues and with the aim of improving the performance of the relaxation method [112]. Khumbar [41] worked on the impact of two types of partitioning according to the method: partitioning on Voltage node or partitioning on Current node. He shows that it has no impact in the classical waveform relaxation method whereas the optimized waveform relaxation converges faster with the partitioning on Voltage node.

1.5.1.3 Graph based partitioning

We have seen a physical point of view of partitioning, a point of view which will be essential in the precision of an EMT-TS co-simulation, but let us remember that one of the objectives of this co-simulation is to save time and therefore ideally to simulate in parallel. A good partitioning is essential for parallel computation, indeed the quality of partitioning has an impact on the speed of computation (for load balance considerations) but also on the good convergence of the algorithm of parallel computation as we will see in the next chapters. The bases of this partitioning must be a balance of the loads allocated to each processor as well as a minimization of the communication costs.

The electrical network can be seen as a graph (by perceiving the components as vertices and the lines connecting these components as edges) and therefore a graph partitioning tool can be used to fairly share the computational loads between the processors. An example of a graph partitioning tool is ParMETIS [90, 59], which is a library for partitioning an unstructured graph, minimizing as much as possible the number of edges of the graph cut by partitioning. This tool is therefore based on the network connectivity.

Aristidou [4] pointed out that the partitioning of the electrical network depends on multiple factors such as the size and type of the network considered, communication costs, and even the intrinsic dynamics of the system. For example, it is better not to cut in a too dense place (for example not to cut a component) to avoid this, a dependence matrix \mathbf{D} can be constructed, if the system has N equations and N unknown, the size \mathbf{D} will be $N \times N$ and $\mathbf{D}(\mathbf{i}, \mathbf{j}) = \mathbf{1}$ if the \mathbf{j} -th variable plays a role in the \mathbf{i} -th equation, 0 otherwise.

Falcao & al [36] performed a partitioning based on the lines in order to take advantage of the delay induced by them, but then taking into account the load balance. They underlined the fact that a physical partitioning of the system may not be optimal. Hence, by cutting with respect to the line, there will certainly be processors whose computational load will be much heavier than the others, and therefore the least loaded will have to idle while waiting for the others to finish their calculations, which is not optimal since the capacity of the machine is not fully mobilized. The Falcao & al 's technique is once the network is cut on the lines, the lines are

prioritized, so the importance of the connections is evaluated, the subnets with the weakest connections are redistributed between the less loaded processors (this technique is effective especially where there is more sub-domain than processors). Shu & al [107] built a partitioning algorithm whose main objective is to maintain an equal calculation weight between the sub-parts and to minimize the connections between the different sub-networks. First, network partitioning is treated as a graph partitioning problem. A function for evaluating the design weight to be minimized is constructed. $\text{Computation-weight-to-be-minimized} = \text{Computation weight of the heaviest sub-network (in computation weight)} + \text{Total weight of communications between sub-networks}$. In a second step, the entire network is considered to be an assembly of different regions (whose communication between regions is less close than within a region) or province or station depending on the size wanted for the subnetworks. What was previously obtained with the partitioning of the graph will be refined by putting the weights in relation to the strength of the connections observed. At each refinement the $\text{Computation-weight-to-be-minimized}$ function is re-evaluated and when the result exceeds that initially obtained (after the first partitioning considering the problem only as a graph partitioning), then the algorithm stops and the preserved partitioning is the penultimate (the one before the initial weight is exceeded).

We have seen that it is important that in addition to the physical aspect, a more practical aspect, concerning the machines used, is taken into account to be effective. These approaches allow these two partitioning points of view to be linked. However, it should not be forgotten that they are used on a domain using only one type of simulation, so the calculation weights of the sub-networks can be comparable, and there are no constraints related to the components to be simulated in more detail than the others.

1.5.2 The interaction protocol

To co-simule several simulation types, it is necessary to orchestrate data exchange and define each simulation executed time. This orchestration is called iteration protocol, it can be implemented different ways, an architecture master/slave type is possible or the two simulations can run independently. The iteration protocol plays a key role in the accuracy and speed of the simulation. There are then two possibilities: serial protocols or parallel protocols, although the architecture of the computing machine must be taken into account in the choice, we will limit ourselves to cases where we use machines using MIMD type architectures. During serial protocols, one simulator operates while the other one is idle, waiting to retrieve information from the one running, in order to update and/or make new predictions. A good point of this type of simulation protocol is that, if the partitioning allows the co-simulation to be convergent, the EMT part often requires only one iteration

thanks to the small size of its time-steps (unless there are non-linearities) [55]. An example of a serial protocol will be a multiplicative schwarz used by Plumier & al [95]. One of the main objectives of co-simulation is to gain speed, which is why parallel protocols will be preferred.

In the case of heterogeneous EMT-TS co-simulation, the protocol must also take into account the difference in time step, most often the EMT time step is chosen so that $\Delta t_{emt} = m * \Delta t_{ts}$ with $m \in \mathbb{N}$, and the macro time step is often chosen to be the TS time step. The EMT subsystem can then be updated at each EMT time step with interpolated values.

During a parallel protocol, the two simulators operate at the same time and exchange data during the appointment time (which is often chosen at the end of the TS simulation time step). The same time step is replayed until the convergence of exchanged datas. The protocol can also adapt itself to the state of the circuit, during a disturbance, the EMT system can, for example, be updated by the TS subsystem to scheduled appointments, but does not transmit information to the TS subsystem until the disturbance is smoothed, it's also possible to iterate only on certain parts and give the information to TS system only at the end of the process [113].

The protocol can change practically at each stage of the simulation. Shu & al [106] set up a mathematical index taking into account the evolution of the current at the interface between the two simulations from one time step to the next. If the current has changed a lot during this period, it considers that the dynamics are fast, then for the next time step, a serial protocol will be preferred, because the serial protocol seems more precise, otherwise a parallel protocol will be preferred. In co-simulation algorithms such as non-iterative Jacobi, zero-order hold iterative co-simulation and non-iterative algorithm improving variables smoothing, the delay of one co-simulation step (i.e. TS time step delay) between the given inputs and the retrieved outputs of the TS and EMT systems can lead to instabilities. Some iterative techniques such as the fixed point method [31] or the Newton-like method [95, 1] can, even with a high order smoothing constraints, solve the so-called "constraint function" corresponding to the interface of the systems [32].

1.5.3 Exchanging Values

During a co-simulation, an important axis is the choice of the values to be exchanged. A classic choice is the VI method where one model sends voltages and receives currents and the other model sends currents and receives voltages. An acceptable choice is the power-conjugate interface (exchanging values whose product gives power, the VI method is so a power-conjugate interface), this choice allows to have more stable and more precise results in the presence of delays. Thus, Rimorov & al focused on proposing power-conjugate interface that combines current

and voltage through an "impedence" parameter resulting from non-physical related boundary conditions [102].

1.5.3.1 Transformation of the values

For EMT-TS co-simulation, whether it is an EMT-Phasor or EMT-Dynamique Phasor co-simulation, the datas to be exchanges must be transformed so that it respects the representation of the values in the other subdomain.

To extract the phasor signal from an EMT signal, a commonly used method is to extract the amplitude using the root mean square (RMS) of the waveform and the angle is obtained using the fundamental value of the waveform [15],[92]. To perform this extraction, a window the size of one cycle of the fundamental period of the signal is necessary. This is the major drawback of using a method based on the Fourier transform. Another difficulty when transforming values from EMT to dynamic phasors is that all harmonics must be extracted independently, Mudunkotuwa & al [83] proposed a method to extract dynamic phasor without computing all harmonics. A commonly used method that does not introduce any delay is referential switching, Plumier & al [96] supplements this approach with a low pass filter. Other commonly used methods are the curve fitting techniques [94]. For phasor transformation to EMT, linear interpolation is often used [92],[96]. Finally, some co-simulate EMT models with phasor models using a dynamic phasor region as a buffer region, and thus perform EMT-Dynamic Phasor transformation and dynamic Phasor-Phasor transformation [60].

A more developed state of the art on the methods used in this thesis is written in chapter 3.

1.5.3.2 Dynamic equivalent

Often the models need to access an equivalent representation of the other model. In this case, representations updated with information coming from the subsystem that it represents, are created: dynamic equivalents.

Build a dynamic equivalent consists in replacing certain elements or element sets of the complete network with another element in order to simplify the simulation of the network while keeping a great fidelity of the physical behavior of the network (or the piece of network) concerned. The type of equivalent system used depends on the type of phenomenon to be observed. Annakkage [3] classifies the equivalents into three kinds: High frequency equivalent (HFE); Low frequency equivalent; and Wideband equivalent when both low and high frequency transient must be adequately represented.

Two kinds of high frequency equivalent are available: Two Layer Network Equivalent (TLNE) and Frequency Dependent Network Equivalent (FDNE) which is

the HFE the most used in co-simulation EMT-TS. Annakkage reminds that, for HFEs, the subsystems to be modeled are assumed to be linear [3].

To obtain the FDNE of a subsystem, the first step is to obtain the admittance matrix using nodal analysis. The only preserved values of the matrix are those involved in the terminal ports (the ports that interface with the other subsystem), a "terminal" admittance matrix is obtained. Subsequently, a method is used to fit the frequency admittance characteristics in a rational function [97]. The most used methods to construct this rational function are vector fitting, Matrix-Pencil-Method and Loewner Matrix techniques. Of these three methods, the one that gives the most accurate results is the vector fitting method [80]. These three methods have been combined by Morales & al in order to preserve high accuracy of the vector fitting technique while accelerating its convergence [79]. To build the TLNE, the subsystem is divided into two layers: a low order frequency transmission lines model and an FDNE.

Although much effort has gone into finding other equivalent models for EMT-TS co-simulation, most of them are still based on Norton and Thevenin equivalents. The fact that most equivalents are based on Norton and Thevenin equivalents is a major drawback, since this type of equivalent is only valid for linear networks. [55] and [95] provide an overview of the equivalents used in the EMT-TS co-simulation context.

In EMT-TS co-simulation, equivalents are used so that each subsystem has a full representation of the other subsystem. So instead of updating the other simulator directly with boundaries values, a subsystem will update its full representation in the other simulator.

The subsystem modeled in EMT will be represented with an equivalent of low frequency type equivalent in the TS system and The TS system will be represented with an equivalent of high frequency type (such as an FDNE admittance in parallel with an ideal current source [95]) within the EMT simulator (because of frequency which is of interest for the EMT part are the high frequency).

Although most studies on EMT-TS co-simulation are carried out with equivalent models and their usefulness in this type of co-simulation is defended by Jalili-Marandi & al [55], who consider that each simulator requires knowing the characteristics of the other zone. It is legitimate to ask whether this knowledge is necessary or whether it is sufficient for each subsystem to have information on what is happening at its border. For instance, Le-Huy & al [63], do not have an equivalent model in its co-simulation, so equivalent model, in co-simulation EMT-TS, are not a necessity. Indeed, the use of equivalent models requires an additional simulation, as well as additional calculations and new calculations to be carried out in case, for example, of a topology change in a subsystem (moreover this recalculation must be taken into account in the protocol iteration).

1.6 Outline of the thesis & Contributions

In order to obtain an EMT-TS co-simulation using domain decomposition methods, it was a question, initially, of working on the domain decomposition methods applied to the electrical circuit, to choose those which seemed to us the more relevant and check that they are working properly within the framework of the electrical networks. Then to gradually modify these methods in order to apply them in a heterogeneous way, it was necessary, among other things, to choose and improve accurate transformation operators, with the constraint that these operators do not modify the properties of the networks allowing to use the domain decomposition methods previously chosen. At each stage, it was checked that the co-simulation combined the advantages of the two types of modeling. Then, it was proven that the chosen domain decomposition method was part of a larger framework, which allowed us to develop different simulation strategies. Finally, a simulation platform based on the methods developed throughout this thesis has been designed and is under development.

This PhD is presented of five separate chapters, the content of each chapter is as follows:

- The first chapter contain the introduction, an explanation of how DAE systems representing electrical circuits are constructed and a state of the art concerning EMT-TS co-simulation.
- The second chapter contains a summary of the Schwarz domain decomposition method. This method is then applied in the case of a homogeneous decomposition of an electric circuit. The convergence of this type of method is studied and Aitken's convergence acceleration method is explained and applied to the electrical circuit. This work was presented at the DD26 conference.
- Chapter three contains the EMT-TS heterogeneous domain decomposition method. To carry out this co-simulation, a modeling using dynamic phasors was chosen for the TS part, this choice is motivated by the fact that the dynamic phasor makes it possible to recover oscillations invisible to the conventional phasor, which can allows one to reduce the part simulated in EMT. The method used in chapter one is modified to be used in the heterogeneous case. Work on the passage of data is carried out. We then show that if the translation operators between models are linear, Aitken's convergence acceleration method still applies. However, the differences in representations and the difference in time step length are a difficulty for the algebraic calculation of the error operator, which is therefore computed numerically. The error operator is calculated only on the TS part of the interface values to overcome

the difficulty of too many values brought into play by the EMT part and thus limit the weight of the computations. This method then makes it possible to accelerate convergence towards the true solution and this even in the cases where the method diverges. Consequently this allows one to ignore the constraint of a contracting error operator in the partitioning. Therefore, the developed method is not sensitive to the partitioning. This makes it possible to address the partitioning problem for EMT-TS co-simulation, a problem that has been widely studied (cf section 1.5.1). This work was presented at the Numdiff16 conference.

- In chapter four, the dynamic iteration method is presented. It is then shown that the method used in previous chapters is a special case of Dynamic Iteration. We proved that the Dynamic Iteration can also be accelerated with Aitken's convergence acceleration method. Two strategies are then presented for using this method: a sequential strategy and a pipelined strategy. The pipelined strategy makes it possible to use the Dynamic Iteration method coupled with the Aitken acceleration technique in circuits comprising non-linear components as well as with variable step simulations. Hence, we show that it is possible to accelerate the convergence of several successive time steps at the same time using the Aitken technique, the development of this pipelined strategy allows to simulate networks including nonlinear components without having to perform more iterations. This work was presented at the DD27 conference.
- Chapter five contains a review of the OpenModelica suite of tools. The use of these tools is explored and its current status with its advantages as well as its drawbacks and limitations are discussed. The steps to follow to use the FMI standard with models developed in the OpenModelica tool suite are detailed. Finally, a platform based on the work of the previous chapters and under development is presented. This part was presented at the Electrimacs conference.

Chapter 2

Schwarz method in the homogeneous EMT or TS case

Contents

2.1	Introduction	22
2.2	The Schwarz domain decomposition method	24
2.2.1	Generalized Schwarz alternating method	24
2.2.2	Connection between the Steklov-Poincaré operators and the Schur complement	26
2.3	Schwarz method for electrical circuit DAE	27
2.3.1	Time discretizing of linear electrical circuit	28
2.3.2	RAS splitting in Detail	29
2.3.3	RAS algorithm	29
2.4	Error operator and Acceleration of convergence	30
2.4.1	The RAS error operator	31
2.4.2	The Aitken's acceleration of convergence of the RAS	32
2.4.3	Acceleration of the RAS in the electrical circuit context	34
2.5	Numerical results	36
2.6	Conclusion	42

2.1 Introduction

As presented in the previous chapter the modeling of an electrical circuit can be represented mathematically by a system of differential and algebraical equations where the differential part comes from the different components used to model inductance and capacitance phenomena, the algebraic part comes from Kirchhoff's laws of current (Kirchhoff's point rule) and of voltage (Kirchhoff's loop rule). Let us consider the DAE system that follows where, for all $t \in [0, T]$, $x(t) \in \mathbb{R}^{n_1}$ are the differential unknowns and $y(t) \in \mathbb{R}^{n_2}$ the algebraic ones:

$$\begin{cases} F(t, \dot{x}, x, y) & = 0 \\ g(t, x, y) & = 0 \\ x(0) & = x_0 \in \mathbb{R}^{n_1} \\ y(0) & = y_0 \in \mathbb{R}^{n_2} \end{cases} \quad (2.1)$$

The variables x and y can represent voltage or current depending on the electrical components involved in the electrical circuit. x_0 (respectively y_0) is the initial value of x at time $t = 0$ (respectively y). The function $F : \mathbb{R} \times \mathbb{R}^{n_1} \times \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \rightarrow \mathbb{R}^{n_1}$ gathers the equations of the DAE system involving some derivatives of unknowns x and the function $g : \mathbb{R} \times \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \rightarrow \mathbb{R}^{n_1}$ gathers the equations of the DAE without derivate of unknowns and correspond to the algebraical constraints as for example the Kirchhoff's laws of current.

Our goal is to solve this DAE system by splitting it into several parts (at least two), i.e. splitting the set of unknowns into several subsets by gathering the differential or algebraic equations associated with the unknowns belonging to the same subset. For example, if we consider two subsets the original DAE system will be split into two DAE systems to be solved on the time interval $[T_n^+, T_{n+1}^-]$:

$$\begin{cases} F_1(t, \dot{x}_1, x_1, y_1, \tilde{x}_2, \tilde{y}_2) & = 0 \\ g_1(t, x_1, y_1, \tilde{x}_2, \tilde{y}_2) & = 0 \\ x_1(T_n^+) & = x_1^n \\ y_1(T_n^+) & = y_1^n \end{cases} \quad \begin{cases} F_2(t, \dot{x}_2, x_2, y_2, \tilde{x}_1, \tilde{y}_1) & = 0 \\ g_2(t, x_2, y_2, \tilde{x}_1, \tilde{y}_1) & = 0 \\ x_2(T_n^+) & = x_2^n \\ y_2(T_n^+) & = y_2^n \end{cases} \quad (2.2)$$

Where \tilde{x}_2 and \tilde{y}_2 in the DAE subsystem 1 (respectively \tilde{x}_1 and \tilde{y}_1 in the DAE subsystem 2) are representations of the solutions x_2 and y_2 in the DAE subsystem 2 (respectively x_1 and y_1 in the DAE subsystem 1). They can be considered as inputs for the current DAE subsystem and must be updated at some rendez-vous time point T_n^+ during the time simulation.

If $\tilde{x}_2 = x_2$ and $\tilde{y}_2 = y_2$ (respectively $\tilde{x}_1 = x_1$ and $\tilde{y}_1 = y_1$), the two DAE subsystems are said to be strongly coupled and can not be solved separately. Co-simulation techniques consist in having approximations for \tilde{x} and \tilde{y} such as Zero Order Hold (ZOH), where the \tilde{x} and \tilde{y} are frozen at their values at the time of the

previous rendez-vous point. Some other polynomial approximations for \tilde{x} and \tilde{y} such as linear approximations from the values at previous rendez-vous point (First Order Hold) or with polynomials with higher degree (Second Order Hold or Third order Hold) can be used. Some extrapolation techniques with delay such as the $C(p, q, j)$ scheme of [44] where the input values are extrapolated to order j^{th} from the solution's values taken at p regular rendez-vous points in the past also exist.

The major drawback of such approaches, is the difference of the value of the solution of one subsystem and its representation in the other subsystem at the next rendez-vous time point and it also limits the size of the macro time step separating two rendez-vous points times. One solution to avoid this delay at the next rendez-vous point is to consider iterative algorithms. These algorithms consist in iteratively updating the inputs \tilde{x} and \tilde{y} in order to have at the end of the macro-time step the same values of the components x and y that they represent in the subsystems which provide them. This update needs to integrate the subsystems on the macro-time step at each iterate.

Schematically the iterative algorithm on the time interval $[T_n^+, T_{n+1}^-]$ is written as follows: starting from initial inputs $\tilde{x}^{(0)}$, $\tilde{y}^{(0)}$, the algorithm iterates over these values until they no longer change:

$$\left\{ \begin{array}{l} F_1(t, \dot{x}_1, x_1, y_1, \tilde{x}_2^{(k)}, \tilde{y}_2^{(k)}) = 0 \\ g_1(t, x_1, y_1, \tilde{x}_2, \tilde{y}_2) = 0 \\ x_1(T_n^+) = x_1^n \\ y_1(T_n^+) = y_1^n \end{array} \right. , \quad \left\{ \begin{array}{l} F_2(t, \dot{x}_2, x_2, y_2, \tilde{x}_1^{(k)}, \tilde{y}_1^{(k)}) = 0 \\ g_2(t, x_2, y_2, \tilde{x}_1, \tilde{y}_1) = 0 \\ x_2(T_n^+) = x_2^n \\ y_2(T_n^+) = y_2^n \end{array} \right. \quad (2.3)$$

$$H_1(\tilde{x}_1^{(k+1)}, \tilde{y}_1^{(k+1)}, x_1^{n+1}, y_1^{n+1}) = 0, \quad H_2(\tilde{x}_2^{(k+1)}, \tilde{y}_2^{(k+1)}, x_2^{n+1}, y_2^{n+1}) = 0. \quad (2.4)$$

Where functions H_1 and H_2 are constraint functions on the inputs in order to guarantee the same values of the inputs as the values that they represent in the other subsystem at the end of the macro step simulation. The way these constraint functions are satisfied can lead to Newton type algorithms such as IFOSMONDI-JFM [32] or IFOSMONDI fixed-point type algorithm [31]. Each of them has advantages and drawbacks, the most important drawback for the fixed point algorithm is its non-contractant property leading in some cases to a non-convergent algorithm.

We focus in this thesis on the special choice of a fixed point algorithm to satisfy the inputs constraint functions, namely the Schwarz type domain decomposition method [105]. We first recall the Schwarz algorithm and some of its variants [33] [23] and then we apply it to the system of DAE arising from electrical circuit simulation.

2.2 The Schwarz domain decomposition method

2.2.1 Generalized Schwarz alternating method

The continuous Schwarz method was introduced by the german analyst Hermann Schwarz at the end of the 19th century [105]. He first invented this method to solve the Poisson problem with Dirichlet boundary conditions on a complex geometry of a rectangle covering a disk in its middle. For this geometry, there is no analytical solution unlike the cases of the rectangle and the disk. Then he proposed and proved the convergence of the algorithm consisting in solving the problem alternately on the disk then on the rectangle by taking the values of the other domain as new boundary conditions for the local problem to be solved. Then, the algorithm iterates until the boundary conditions no longer move. Boundary conditions act like the \tilde{x} and \tilde{y} variables in our DAE subsystems.

Schwarz domain decomposition method in space applied to system of partial derivative equations with some elliptic terms has been widely studied during the past years. Especially with the developpement of large parallel computing ressources, it shows to be a well-adapted algorithm for such architecture. It appears that the choice of boundary conditions can impact strongly the convergence of the Schwarz type algorithm [33] [43] [40].

The generalized alternating Schwarz method proposed by Engquist and Zao [33] allows to generalize several Schwarz techniques [99] to solve a uniformly elliptic second order scalar boundary problem on a domain Ω with Dirichlet boundary conditions on $\Gamma = \partial\Omega$:

$$\begin{cases} \mathbb{L}(z)u(z) &= f(z), \text{ for } z \in \Omega, \\ \gamma_0 u(z) &= g(z), \text{ for } z \in \Gamma, \end{cases} \quad (2.5)$$

Where $f \in H^{-1}(\Omega)$, $g \in H^{1/2}(\Gamma)$, γ_0 is the trace operator and the partial derivative operator $L(z)$, $z \in \Omega$ is given by $L(z)u(z) = -\sum_{i=1}^n \sum_{j=1}^n \frac{\partial}{\partial x_j} [a_{ji}(z) \frac{\partial}{\partial z_i} u(z)]$

with $a_{ji} = a_{ij} \in L^\infty(\Omega)$, $i, j = 1, \dots, n$. $L(\cdot)$ is assumed to be uniformly elliptic i.e.: $\exists c_0 > 0$, independent of z such that, $\forall z \in \Omega$:

$$\sum_{j=1}^n \sum_{i=1}^n a_{ji}(z) \xi_j \xi_i \geq c_0 \cdot |\xi|^2, \quad \forall \xi \in \mathbb{R}^n$$

For simplicity, let us consider the case where Ω is decomposed into two subdomains Ω_1 and Ω_2 , overlapping or not and generating two artificial boundaries Γ_1 , Γ_2 . We define $\bar{i} = \text{mod}(i, 2) + 1$ and $\Omega_{11} = \Omega_1 \setminus \Omega_2$, $\Omega_{22} = \Omega_2 \setminus \Omega_1$, $\Omega_{12} = \Omega_1 \cap \Omega_2$, $\Omega_{11} = \Omega_1 \setminus \Omega_{12}$, $\Omega_{22} = \Omega_2 \setminus \Omega_{12}$ if there is overlap. The GSAM method solves until convergence the problem (2.5) restricted to the subdomain Ω_1 (respectively Ω_2) with boundary conditions on $\partial\Omega_1$ (respectively $\partial\Omega_2$) and generalized boundary conditions $\Lambda_1 u_1 + \lambda_1 \frac{\partial u_1}{n_1} = \mu_2$ on Γ_1 (respectively $\Lambda_2 u_2 + \lambda_2 \frac{\partial u_2}{n_2} = \mu_1$), where Λ_i

are operators and λ_i are constants.

The value $\mu_2 = \Lambda_1 u_2 + \lambda_1 \frac{\partial u_2}{\partial n_1}$ (respectively $\mu_1 = \Lambda_2 u_1 + \lambda_2 \frac{\partial u_1}{\partial n_2}$) is defined with respect to the solution u_2 in Ω_2 (respectively u_1 in Ω_1). As the solution on the subdomains Ω_i are unknown, we iterate the resolution, taking the values of the generalized boundary conditions μ_i from the solution obtained at the previous iteration in Ω_i .

Algorithm 1 GSAM: Multiplicative ($M_1 = N_2 = (2k + 1), M_2 = (2k + 2), N_1 = (2k)$), Additive ($M_1 = M_2 = (k + 1), N_1 = N_2 = (k)$)

- 1: DO until convergence
- 2: Solve

$$L(z)u_1^{M_1}(z) = f(z), \forall z \in \Omega_1, \quad (2.6)$$

$$u_1^{M_1}(z) = g(z), \forall z \in \partial\Omega_1 \setminus \Gamma_1, \quad (2.7)$$

$$\Lambda_1 u_1^{M_1} + \lambda_1 \frac{\partial u_1^{M_1}(z)}{\partial n_1} = \Lambda_1 u_2^{N_1} + \lambda_1 \frac{\partial u_2^{N_1}(z)}{\partial n_1}, \forall z \in \Gamma_1 \quad (2.8)$$

- 3: Solve

$$L(z)u_2^{M_2}(z) = f(z), \forall z \in \Omega_2, \quad (2.9)$$

$$u_2^{M_2}(z) = g(z), \forall z \in \partial\Omega_2 \setminus \Gamma_2, \quad (2.10)$$

$$\Lambda_2 u_2^{M_2} + \lambda_2 \frac{\partial u_2^{M_2}(z)}{\partial n_2} = \Lambda_2 u_1^{N_2} + \lambda_2 \frac{\partial u_1^{N_2}(z)}{\partial n_2}, \forall x \in \Gamma_2. \quad (2.11)$$

- 4: End DO
-

The additive version of GSAM consists in solving the problems on the subdomains simultaneously during an iteration whereas for the multiplicative version, the problems on the subdomains are solved successively one after the other to update the μ_i values.

Depending on the specific choice of the operators Λ_i and the values of the scalars λ_i , we obtain in the table 2.1 the families of Schwarz domain decomposition techniques. If $\Lambda_1 = \Lambda_2 = Id$ and $\lambda_1 = \lambda_2 = 0$ then the multiplicative version is the classical version on multiplicative Schwarz. If $\Lambda_1 = \Lambda_2 = constant > 0$ and $\lambda_1 = \lambda_2 = 1$ then it is the modified Schwarz method proposed by P.-L. Lions in [67].

Overlap	Λ_1	Λ_2	λ_1	λ_2	Method
yes	Id	Id	0	0	Schwarz
yes	Id	Id	α	α	ORAS [109]
No	Id	0	0	1	Neumann-Dirichlet [73]
No	Id	Id	1	1	Modified Schwarz [66]

Table 2.1: Derived methods [109, 73, 66] obtained from the specific choice of operators Λ_i and scalar values λ_i in the GSAM method.

Engquist and Zao [33] show that If Λ_1 (or Λ_2) is the Dirichlet to Neumann operator mapping in correspondence of the artificial interface Γ_1 (or Γ_2) for the homogeneous partial differential equation (or Γ_2) for the homogeneous partial differential equation in Ω_2 (or Ω_1) with edge conditions homogeneous on $\partial\Omega_2 \cap \partial\Omega$ (or $\partial\Omega_1 \cap \partial\Omega$), then the generalized Schwarz Alternate method converges in two steps. These operators are global operators (linking all subdomains together). In practice, algebraic approximations of these operators are used (see [108] [85] [84] [42] [47]).

2.2.2 Connection between the Steklov-Poincaré operators and the Schur complement

As we see, the Dirichlet to Neumann mapping operator plays an essential role in the convergence of the Schwarz type method. In the electrical circuit simulation context, it can be difficult to exhibit some co-normal derivative to a boundary, as the boundaries are usually limited to isolated connecting points. Nevertheless, we can consider at the discrete level a connection between the Schur complement and the Steklov-Poincaré operator.

For simplicity, we consider two subdomains without overlap. Then the discretization of Eq (2.5), with finite elements, finite volumes or finite differences, leads to the following matrix system:

$$Au = \begin{pmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33}^{(1)} + A_{33}^{(2)} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3^{(1)} + f_3^{(2)} \end{pmatrix} \quad (2.12)$$

Where u_1 and u_2 are the vectors of the interior unknowns in the subdomains Ω_1 and Ω_2 , respectively, and u_3 representing the vector of the remaining unknowns defined on Γ . The internal unknowns can be formally eliminated from Eq (2.12) by writing:

$$\begin{aligned} u_1 &= A_{11}^{-1}(f_1 - A_{13}u_3), \\ u_2 &= A_{22}^{-1}(f_2 - A_{23}u_3), \end{aligned} \quad (2.13)$$

and by noting

$$\begin{aligned} S_1 &= A_{33}^{(1)} - A_{31}A_{11}^{-1}A_{13}, g_1 = f_3^{(1)} - A_{31}A_{11}^{-1}f_1, \\ S_2 &= A_{33}^{(2)} - A_{32}A_{22}^{-1}A_{23}, g_2 = f_3^{(2)} - A_{32}A_{22}^{-1}f_2, \end{aligned} \quad (2.14)$$

We obtain the Schur complement system

$$Su_3 = (S_1 + S_2)u_3 = g_1 + g_2 = g \quad (2.15)$$

We have the following identity (see [86]):

$$\hat{A}_{11}^{-1} \begin{pmatrix} 0 \\ S_1x_3 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{13} \\ A_{31} & A_{33}^{(1)} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ S_1u_3 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{13} \\ 0 & I \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ u_3 \end{pmatrix} \quad (2.16)$$

The matrix \hat{A}_{11} corresponds to the discretization matrix of the problem on the subdomain Ω_1 with homogeneous Dirichlet boundary conditions on $\partial\Omega_1 \setminus \Gamma$ and Neumann boundary conditions on the interface Γ . Thus, the matrix-vector product S_1x_3 is simply the discrete co-normal derivative on Γ associated to the Eq(2.5) problem on the subdomain Ω_1 with homogeneous Dirichlet conditions on $\partial\Omega_1 \setminus \Gamma$ and Dirichlet conditions prescribed by u_3 on Γ (i.e. for S_2u_3). These matrix-vector products are the discrete equivalent of the Steklov-Poincaré operator on their respective domain.

2.3 Schwarz method for electrical circuit DAE

For the DAE (2.1) problem resulting from a linear electrical circuit, the domain is composed of the voltage and current unknowns which are linked by the properties of the components and Kirshoff's law.

Two strategies can be performed to split (2.1). The first one is to consider the discretization of (2.1), then to split the unknowns with respect to the dependencies of the data in the discrete equations, in this case certain unknowns belonging to the same component can be dispatched on different subdomains. The second strategy consists in dividing the domain into components, with this strategy the unknowns of the same component all belong to the same subdomain. In this chapter, we consider the first strategy. We consider a linear electrical circuit in the following, non linear electrical circuits could be linearized with a state space representation and treated as a linear circuit over the time step. Let us rewrite the linear DAE (2.1) in its state space representation:

$$\begin{cases} \mathbb{I}\dot{x}(t) + Ax(t) + By(t) &= G_1(t), x(0) = x_0, \\ Cx(t) + Dy(t) &= G_2(t), t \in [0, T]. \end{cases} \quad (2.17)$$

Where $x(t) \in \mathbb{R}^{n_1}$ and $y(t) \in \mathbb{R}^{n_2}$ for all $t \in [0, T]$, D is a $n_2 \times n_2$ nonsingular matrix, A and \mathbb{I} are $n_1 \times n_1$ matrix, \mathbb{I} matrix can be the identity or matrix composed of 1s and 0s depending on whether the x variables contain voltages or potentials. B is an $n_1 \times n_2$ matrix, C is an $n_2 \times n_1$ matrix, $G_1(t) \in \mathbb{R}^{n_1}$ and $G_2(t) \in \mathbb{R}^{n_2}$ are known input functions, as the DAE system is representing an electrical network, $G_1(t)$ and $G_2(t)$ are sources vector. Finally, $x_0 \in \mathbb{R}^{n_1}$ is a consistent initial value. Let $n = n_1 + n_2$.

We will condense this system of equations to obtain the form that we will used throughout this work. First, we define the matrix $\mathbb{A} = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ corresponding to the linear operator of the DAE and we define $z(t) = [x(t), y(t)]^T$,

$$G(t) = [G_1(t), G_2(t)]^T \text{ and } \mathbb{I}_d = \begin{pmatrix} \mathbb{I}_{n_1} & 0_{n_1 \times n_2} \\ 0_{n_2 \times n_1} & 0_{n_2 \times n_2} \end{pmatrix}.$$

Then we can rewrite Eq (2.17) as:

$$\mathbb{I}_d \dot{z}(t) + \mathbb{A}z(t) = G(t), \quad x(0) = x_0, \quad t \in [0, T]. \quad (2.18)$$

2.3.1 Time discretizing of linear electrical circuit

We use a backward Euler for time discretization for (2.1), other backward differences formula (BDF) schemes would give similar results with more complicated formula. Let us notice that the trapezoidal rule integration is more often used with the CDA method [30, 74] and variants [103] using two halved time-step Euler backward integrations from the discontinuity detection time-point in case of presence of switches in the network. The important point here is that we have an implicit time integration method.

In the EMT case:

$$\underbrace{\begin{pmatrix} \mathbb{I}_{emt} - \Delta t_{emt} A_{emt} & \Delta t_{emt} B_{emt} \\ C_{emt} & D_{emt} \end{pmatrix}}_{\mathbb{A}_{emt}} \underbrace{\begin{pmatrix} x_{emt}^{n+1} \\ y_{emt}^{n+1} \end{pmatrix}}_{z_{emt}^{n+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_{emt} & 0 \\ 0 & 0 \end{pmatrix}}_{\Theta_{emt}} \underbrace{\begin{pmatrix} x_{emt}^n \\ y_{emt}^n \end{pmatrix}}_{z_{emt}^n} + \begin{pmatrix} \Delta t_{emt} G_{1,emt}^{n+1} \\ G_{2,emt}^{n+1} \end{pmatrix} \quad (2.19)$$

In the Dynamic Phasor case, the equations must first be adapted to the shape of the dynamic phasor: take into account the differentiation property of dynamic phasors (see section 1.3 of introduction) and multiply the number of equations by the number of harmonics that we have chosen to keep. We have also chosen to solve the real and imaginary parts separately. Thus a form of discretized state space for the dynamic phasor is:

$$\underbrace{\begin{pmatrix} \mathbb{I}_{ts} - \Delta t_{ts} A_{ts} & \Delta t_{ts} B_{ts} \\ C_{ts} & D_{ts} \end{pmatrix}}_{\mathbb{A}_{ts}} \underbrace{\begin{pmatrix} x_{ts}^{N+1} \\ y_{ts}^{N+1} \end{pmatrix}}_{z_{ts}^{N+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_{ts} & 0 \\ 0 & 0 \end{pmatrix}}_{\Theta_{ts}} \underbrace{\begin{pmatrix} x_{ts}^N \\ y_{ts}^N \end{pmatrix}}_{z_{ts}^N} + \begin{pmatrix} \Delta t_{ts} G_{1,ts}^{N+1} \\ G_{2,ts}^{N+1} \end{pmatrix} \quad (2.20)$$

EMT and TS cases are quite similar and need to solve a linear system for each time step. In the the following, we drop the superscripts *emt* or *ts* as we treat partitions with homogeneous modeling.

2.3.2 RAS splitting in Detail

We consider the matrix $\mathbb{A} \in \mathbb{R}^{n \times n}$ having a non-zero pattern and the associated graph $G = (W, F)$, where the set of vertices $W = \{1, \dots, n\}$ represents the n unknowns and the set of edges $F = \{(i, j) | (\mathbb{A}_{i,j}) \neq 0\}$ represents the pairs of vertices that are coupled by a non-zero element in \mathbb{A} . Then, we assume that a graph partitioning was applied and that resulted in N non-overlapping subsets W_i^0 whose union is W . Let W_i^p be the p -overlap partition of W , obtained by including all the vertices immediately neighboring the vertices of W_i^{p-1} . Let $W_{i,e}^p = W_i^{p+1} \setminus W_i^p$. Then let $R_i^p \in \mathbb{R}^{n_i \times n}$ ($R_{i,e}^p \in \mathbb{R}^{n_{i,e} \times n}$ and $\tilde{R}_i^0 \in \mathbb{R}^{n_i \times n}$ respectively) be the operator which restricts $w \in \mathbb{R}^n$ to the components of w belonging to W_i^p ($W_{i,e}^p$ and W_i^0 respectively, and the operator $\tilde{R}_i^0 \in \mathbb{R}^{n_i \times n}$ puts 0 to the unknowns belonging to $W_i^p \setminus W_i^0$). Then we defined the local operators $\mathbb{A}_{i,emt} = R_i^p \mathbb{A} R_i^{pT}$ and $\mathbb{E}_{i,e,emt} = R_{i,e}^p \mathbb{A} R_{i,e}^{pT}$.

The DAE system is for a domain W_i^p , integrated between t^n to t^{n+1} :

$$\underbrace{\begin{pmatrix} \mathbb{I}_i - \Delta t A_i & \Delta t B_i \\ C_i & D_i \end{pmatrix}}_{\mathbb{A}_i} \underbrace{\begin{pmatrix} x_i^{n+1} \\ y_i^{n+1} \end{pmatrix}}_{z_i^{n+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_i & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,i}} \underbrace{\begin{pmatrix} x_i^n \\ y_i^n \end{pmatrix}}_{z_i^n} + \underbrace{\begin{pmatrix} \mathbb{I}_{ie} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,ie}} \underbrace{\begin{pmatrix} x_{ie}^n \\ y_{ie}^n \end{pmatrix}}_{z_{ie}^n} - \underbrace{\begin{pmatrix} E_{ie}^A & E_{ie}^B \\ E_{ie}^C & E_{ie}^D \end{pmatrix}}_{\mathbb{E}_{ie}} \underbrace{\begin{pmatrix} x_{i,e}^{n+1} \\ y_{i,e}^{n+1} \end{pmatrix}}_{z_{i,e}^{n+1}} + \underbrace{\begin{pmatrix} \Delta t G_{1i}^{n+1} \\ G_{2i}^{n+1} \end{pmatrix}}_{G_i^{n+1}}. \quad (2.21)$$

The term $\mathbb{I}_{d,ie} z_{ie}^n$ is coming from the fact that differential terms on the interface unknowns can be involved due to the splitting. The terms at time t^n and the source term G_i^{n+1} can be gather in a term b_i^{n+1} independent of the solution z_i^{n+1} . Then the DAE system for the domain W_i^p integrated between t^n and t^{n+1} :

$$\mathbb{A}_i z_i^{n+1} = b_i^{n+1} - \mathbb{E}_{ie} z_{i,e}^{n+1} \quad (2.22)$$

2.3.3 RAS algorithm

The Eq (2.22) can not be directly solved as it needs the value of the solution at $n + 1$ from the others partitions.

Definition 1. The $(k+1)^{th}$ iteration of the Restrictive Additive Schwarz algorithm in the discrete case is written locally for the W_i^p partition and with integrating between t^n and t^{n+1} , if \mathbb{A}_i invertible, as:

$$z_i^{n+1,(k+1)} = \mathbb{A}_i^{-1}(b_i^{n+1} - \mathbb{E}_{ie} z_{i,e}^{n+1,(k)}) \quad (2.23)$$

Remark 1. The term $z_{i,e}^{n+1,(k)}$ can be considered analogous to the Dirichlet conditions, indeed if a differentiated part of the interface conditions exists, it is passed into the term b_i^{n+1} and it is independent of the Schwarz iterations and has no impact on the interface conditions sought at time t^{n+1} .

The Eq (2.23) considers the additive version of the RAS algorithm, where all the partitions iterate the $(k+1)^{th}$ iteration with interface condition taken in the other partition at the $(k)^{th}$ iteration. One can also consider the multiplicative version of the RAS where one partition is solved one by one taking the latest $(k+1)^{th}$ values of the interface available. Algorithm 2 summarizes these two versions of RAS to integrate the DAE from t^n to t^{n+1} for the domain W splitted into two partitions:

Algorithm 2 RAS: Multiplicative ($M_1 = N_2 = (2k+1)$, $M_2 = (2k+2)$, $N_1 = (2k)$), Additive ($M_1 = M_2 = (k+1)$, $N_1 = N_2 = (k)$)

- 1: DO until convergence
- 2: Solve

$$z_1^{n+1,M_1} = \mathbb{A}_1^{-1}(b_1^{n+1} - \mathbb{E}_{1e} z_{1,e}^{n+1,N_1})$$

- 3: Solve

$$z_2^{n+1,M_2} = \mathbb{A}_2^{-1}(b_2^{n+1} - \mathbb{E}_{2e} z_{2,e}^{n+1,N_2})$$

- 4: Enddo
-

2.4 Error operator and Acceleration of convergence

As the Schwarz method can be considered as a fixed-point algorithm, its convergence is not guaranteed. In the PDE with elliptic part case, we have the benefit that the convergence increases with the size of the overlap. We are going to exhibit the RAS matrix operator error on the interface values. Then, in the case of a linear electric circuit, we will take advantage of the non-dependence of the error

operator on the RAS iteration to accelerate the RAS algorithm towards the true solution, whether the RAS is convergent or divergent.

2.4.1 The RAS error operator

We will expose the matrix error operator for the RAS. For this, we first show that the RAS can be seen as a preconditioned iterative Richardson method

Definition 2 (The RAS preconditioning matrix). *With the notation of the RAS method introduced in section 2.3.2, we define the RAS preconditioning matrix M_{RAS}^{-1} as:*

$$M_{RAS}^{-1} \stackrel{def}{=} \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \quad (2.24)$$

Proposition 1 (RAS as a preconditioned Richardson iterative method [39]). *The RAS method can be seen as a preconditioned Richardson iterative method with the preconditioning matrix M_{RAS}^{-1}*

$$z^{n+1,(k+1)} = z^{n+1,(k)} + M_{RAS}^{-1}(\tilde{b}^{n+1} - \mathbb{A}z^{n+1,(k)}). \quad (2.25)$$

Proof. Starting from Eq (2.23), the RAS iteration on the partition W_i^p and the definition of R_i^p and R_{ie}^p :

$$\begin{aligned} z_i^{n+1,(k+1)} &= \mathbb{A}_i^{-1}(b_i^{n+1} - \mathbb{E}_{ie} z_{i,e}^{n+1,(k)}) \\ R_i^p z^{n+1,(k+1)} &= \mathbb{A}_i^{-1}(R_i^p b^{n+1} - \mathbb{E}_{ie} R_{ie}^p z^{n+1,(k)}) \end{aligned}$$

Then we add the contribution of each partition W_i^p and we use the definition of $\mathbb{E}_{ie} = R_i^p \mathbb{A} R_{ie}^{pT}$:

$$\begin{aligned} \sum_{i=0}^{N-1} \tilde{R}_i^{0T} R_i^p z^{n+1,(k+1)} &= \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \tilde{b}^{n+1} - \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \mathbb{A} R_{ie}^{pT} R_{ie}^p z^{n+1,(k)} \\ z^{n+1,(k+1)} &= M_{RAS}^{-1} \tilde{b}^{n+1} - \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \mathbb{A} R_{ie}^{pT} R_{ie}^p z^{n+1,(k)} \end{aligned}$$

As $R_i^p \mathbb{A} = R_i^p \mathbb{A} (R_i^{pT} R_i^p + R_{i,e}^{pT} R_{i,e}^p)$

$$\begin{aligned} z^{n+1,(k+1)} &= M_{RAS}^{-1} \tilde{b}^{n+1} - \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} (R_i^p \mathbb{A} - R_i^p \mathbb{A} (R_i^{pT} R_i^p)) z^{n+1,(k)} \\ &= M_{RAS}^{-1} (\tilde{b}^{n+1} - \mathbb{A} z^{n+1,(k)}) + \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} \mathbb{A}_i R_i^p z^{n+1,(k)} \\ &= z^{n+1,(k)} + M_{RAS}^{-1} (\tilde{b}^{n+1} - \mathbb{A} z^{n+1,(k)}). \end{aligned}$$

□

Definition 3 (The RAS global interface and its restriction operator). *We define the RAS global interface Γ as the concatenation of the $W_{i,e}^p$, i.e $\Gamma = \{W_{0,e}^p, \dots, W_{N-1,e}^p\}$ of size $n_\Gamma = \sum_{i=0}^{N-1} n_{i,e}$. We define the restriction operator R_Γ associated to the RAS global interface as $R_\Gamma = (R_{0,e}^p, \dots, R_{N-1,e}^p)^T \in \mathbb{R}^{n_\Gamma \times n}$. We note $z_\Gamma^{n+1,(k+1)} = R_\Gamma z^{n+1,(k+1)}$*

Proposition 2 (The RAS written on the global interface). *The RAS iterative method can be written as an iterative method involving only the value of iterations on the global interface Γ , i.e there exists an operator $P_\Gamma = R_\Gamma (I - M_{RAS}^{-1} \mathbb{A}) R_\Gamma^T \in \mathbb{R}^{n_\Gamma \times n_\Gamma}$ such that:*

$$z_\Gamma^{n+1,(k+1)} = P_\Gamma z_\Gamma^{n+1,(k)} + c^{n+1}. \quad (2.26)$$

Proof. We have the property that $R_{i,e}^{pT} R_{i,e}^p = R_{i,e}^{pT} R_{i,e}^p R_\Gamma^T R_\Gamma$.

Then we can write: $R_i^p \mathbb{A} R_{i,e}^{pT} R_{i,e}^p R_\Gamma^T R_\Gamma = R_i^p (\mathbb{A} - \mathbb{A} R_{i,e}^{pT} R_{i,e}^p) R_\Gamma^T R_\Gamma$

$$\begin{aligned} z^{n+1,(k+1)} &= M_{RAS}^{-1} \tilde{b}^{n+1} - \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \mathbb{A} R_{i,e}^{pT} R_{i,e}^p z^{n+1,(k)} \\ R_\Gamma z^{n+1,(k+1)} &= R_\Gamma M_{RAS}^{-1} \tilde{b}^{n+1} - R_\Gamma \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} R_i^p \mathbb{A} R_{i,e}^{pT} R_{i,e}^p R_\Gamma^T R_\Gamma z^{n+1,(k)} \\ &= R_\Gamma M_{RAS}^{-1} (\tilde{b}^{n+1} - \mathbb{A} R_\Gamma^T R_\Gamma z^{n+1,(k)}) + R_\Gamma \sum_{i=0}^{N-1} \tilde{R}_i^{0T} \mathbb{A}_i^{-1} \mathbb{A}_i R_i^p R_\Gamma^T R_\Gamma z^{n+1,(k)} \\ &= R_\Gamma z^{n+1,(k)} - M_{RAS}^{-1} \mathbb{A} R_\Gamma^T R_\Gamma z^{n+1,(k)} R_\Gamma z^{n+1,(k)} + R_\Gamma M_{RAS}^{-1} \tilde{b}^{n+1} \\ \underbrace{R_\Gamma z^{n+1,(k+1)}}_{z_\Gamma^{n+1,(k+1)}} &= \underbrace{R_\Gamma (I - M_{RAS}^{-1} \mathbb{A}) R_\Gamma^T}_{P_\Gamma} \underbrace{R_\Gamma z^{n+1,(k)}}_{z_\Gamma^{n+1,(k)}} + \underbrace{R_\Gamma M_{RAS}^{-1} \tilde{b}^{n+1}}_{c^{n+1}}. \end{aligned}$$

□

2.4.2 The Aitken's acceleration of convergence of the RAS

The operator P_Γ involved in Eq (2.26) does not depend on the iteration (k). Then it is possible to accelerate the convergence of the RAS to the true solution.

Proposition 3 (The Aitken acceleration of the RAS). *If there is no eigenvalue equal to 1 in the error operator P of the RAS (i.e the RAS is convergent or divergent) then The RAS iterative method can be accelerated to the true solution $z_\Gamma^{n+1,(\infty)}$ on the interface Γ by the Aitken technique for accelerating the convergence with two consecutive RAS iterations:*

$$z_\Gamma^{n+1,(\infty)} = (I - P_\Gamma)^{-1} (z_\Gamma^{n+1,(k)} - P_\Gamma z_\Gamma^{n+1,(k-1)}), \forall k \geq 1 \quad (2.27)$$

Proof. The true solution satisfies the preconditioned Richardson iterative method:

$$z^{n+1,(\infty)} = z^{n+1,(\infty)} + M_{RAS}^{-1}(\tilde{b}^{n+1} - \mathbb{A}z^{n+1,(\infty)}). \quad (2.28)$$

consequently it also satisfies Eq (2.26):

$$z_{\Gamma}^{n+1,(\infty)} = P_{\Gamma}z_{\Gamma}^{n+1,(\infty)} + c^{n+1},$$

then we can write:

$$z_{\Gamma}^{n+1,(k)} - z_{\Gamma}^{n+1,(\infty)} = P_{\Gamma}(z_{\Gamma}^{n+1,(k-1)} - z_{\Gamma}^{n+1,(\infty)})$$

if 1 is not an eigenvalue of P_{Γ} then $(I - P_{\Gamma})^{-1}$ exists and we obtain:

$$z_{\Gamma}^{n+1,(\infty)} = (I - P_{\Gamma})^{-1}(z_{\Gamma}^{n+1,(k)} - P_{\Gamma}z_{\Gamma}^{n+1,(k-1)}), \forall k \geq 1$$

□

Remark 2. *Once the true solution $z_{\Gamma}^{n+1,(\infty)}$ is obtained on interface Γ , another local resolution is needed to obtain the true solution $z^{n+1,(\infty)}$ on W .*

Remark 3. *We focused on the acceleration by the Aitken's acceleration of the convergence technique for the RAS but we can proceed in the same way for the multiplicative version of the restricted Schwarz.*

Remark 4. *Contrary to other domain decomposition acceleration method where some approximation of the local Steklov-Poincaré operator are used, in the Aitken acceleration of the Schwarz method this is the effect of all local Steklov-Poincaré operators on the convergence that is taken into account. The efficiency of the RAS method accelerated by the Aitken process depends on the cost for computing the operator P which is of the size of n_{Γ}*

In particular, for two partitions that will use in numerical test, the operator P exhibits a sparse structure linking the errors for each local interface to the error on the local interface of the other partition.

Definition 4 (error on the local interface $W_{i,e}^p$). *Let us define $e_{i,e}^{n+1,(k)} \in W_{i,e}^p$ the error between the true solution and the RAS iteration on the local interface:*

$$e_{i,e}^{n+1,(k)} = R_{i,e}R_{\Gamma}^TR_{\Gamma}(z^{n+1,(k)} - z^{n+1,(\infty)}) = (z_{i,e}^{n+1,(k)} - z_{i,e}^{n+1,(\infty)}) \quad (2.29)$$

Proposition 4. *For two partitions case the operator P has a sparse structure with local operators P_i acting on the error on the other local interface such that:*

$$\begin{pmatrix} e_{0,e}^{n+1,(k+1)} \\ e_{1,e}^{n+1,(k+1)} \end{pmatrix} = \begin{pmatrix} 0 & P_0 \\ P_1 & 0 \end{pmatrix} \begin{pmatrix} e_{0,e}^{n+1,(k)} \\ e_{1,e}^{n+1,(k)} \end{pmatrix} \quad (2.30)$$

where

$$P_0 = -R_{0,e}^p \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} \quad (2.31)$$

$$P_1 = -R_{1,e}^p \tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} \quad (2.32)$$

Proof. From Eq (2.23) one can write:

$$\begin{aligned} z_i^{n+1,(k+1)} &= \mathbb{A}_i^{-1} (b_i^{n+1} - \mathbb{E}_{ie} z_{i,e}^{n+1,(k)}) \\ z_i^{n+1,(\infty)} &= \mathbb{A}_i^{-1} (b_i^{n+1} - \mathbb{E}_{ie} z_{i,e}^{n+1,(\infty)}) \\ (z_i^{n+1,(k+1)} - z_i^{n+1,(\infty)}) &= -\mathbb{A}_i^{-1} \mathbb{E}_{ie} (z_{i,e}^{n+1,(k)} - z_{i,e}^{n+1,(\infty)}) \\ \sum_{i=0}^1 \tilde{R}_i^{0T} (z_i^{n+1,(k+1)} - z_i^{n+1,(\infty)}) &= -\sum_{i=0}^1 \tilde{R}_i^{0T} \mathbb{A}_i^{-1} \mathbb{E}_{ie} R_{i,e}^p (z^{n+1,(k)} - z^{n+1,(\infty)}) \\ \tilde{R}_0^{0T} e_0^{n+1,(k+1)} + \tilde{R}_1^{0T} e_1^{n+1,(k+1)} &= -\tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} R_{0,e}^p (\tilde{R}_0^{0T} e_0^{n+1,(k)} + \tilde{R}_1^{0T} e_1^{n+1,(k)}) \\ &\quad -\tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} R_{1,e}^p (\tilde{R}_0^{0T} e_0^{n+1,(k)} + \tilde{R}_1^{0T} e_1^{n+1,(k)}) \end{aligned}$$

Using the properties of \tilde{R}_i^{0T} and $R_{i,e}^p$ for the two partitions case:

$$R_{i,e}^p \tilde{R}_i^{0T} z = 0, \forall z \in W_i^p \quad (2.33)$$

$$R_{j,e}^p \tilde{R}_i^{0T} e_i = e_{j,e}, i \neq j, \forall e_i \in W_i^p \quad (2.34)$$

We obtain the P_0 and P_1 matrices of the P operator:

$$\begin{aligned} e_{0,e}^{n+1,(k+1)} &= -R_{0,e}^p \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} e_{1,e}^{n+1,(k)} = P_0 e_{1,e}^{n+1,(k)} \\ e_{1,e}^{n+1,(k+1)} &= -R_{1,e}^p \tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} e_{0,e}^{n+1,(k)} = P_1 e_{0,e}^{n+1,(k)} \end{aligned}$$

□

2.4.3 Acceleration of the RAS in the electrical circuit context

As part of the time simulation of an electrical circuit, the acceleration of the RAS method can benefit from certain opportunities:

If the circuit is linear and the time integrator uses a constant time step size, the state space matrix does not change. Therefore, the error operator P is the same for all time steps and it suffices to calculate the error operator for the first time step. It will suffice to carry out one single RAS iteration at each time step to obtain the converged solution using the P operator calculated at the first time step, the initial value $z^{n+1,(0)}$ and the iteration $z^{n+1,(1)}$.

If the state space matrix change due to a nonlinearity or the reconfiguration of the network for example, then it is necessary to recalculate P .

Remark 5. For small circuits, the additional local resolution after speeding up RAS convergence on solution interfaces can be avoided by constructing the global error operator P_g working on the entire partitions. For example for the case of the two partitions the global error operator P_g is written (see Eq. (2.33) in proposition 4):

$$P_g = -\tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} R_{0,e}^p - \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} R_{1,e}^p \quad (2.35)$$

Proposition 5. Let $\Lambda(P)$ (respectively $\Lambda(P_g)$) the set of non zero eigenvalues of the error operator P (respectively P_g). We have the property:

$$\Lambda(P) = \Lambda(P_g) \quad (2.36)$$

Proof. Let $w \in W$ an eigenvector associated to the eigenvalue λ of P_g . Then

$$\begin{aligned} P_g w &= -\tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} R_{0,e}^p w - \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} R_{1,e}^p w = \lambda w \\ R_{0,e}^p P_g w &= -R_{0,e}^p \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} R_{1,e}^p w = P_1 w_{1,e} = \lambda w_{0,e} = \lambda R_{0,e}^p w \\ R_{1,e}^p P_g w &= -R_{1,e}^p \tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} R_{0,e}^p w = P_0 w_{0,e} = \lambda w_{1,e} = \lambda R_{1,e}^p w \\ \begin{pmatrix} 0 & P_0 \\ P_1 & 0 \end{pmatrix} \begin{pmatrix} w_{0,e} \\ w_{1,e} \end{pmatrix} &= \lambda \begin{pmatrix} w_{0,e} \\ w_{1,e} \end{pmatrix}, \lambda \in \Lambda(P_g) \Rightarrow \lambda \in \Lambda(P) \end{aligned} \quad (2.37)$$

Conversely, Let $\begin{pmatrix} w_{0,e} \\ w_{1,e} \end{pmatrix} \in \Gamma$ an eigenvector of P associated to the eigenvalue λ .

$$\begin{aligned} P_1 w_{1,e} &= \lambda w_{0,e} = -R_{0,e}^p \tilde{R}_1^{0T} \mathbb{A}_1^{-1} \mathbb{E}_{1e} w_{1,e} \\ P_0 w_{0,e} &= \lambda w_{1,e} = -R_{1,e}^p \tilde{R}_0^{0T} \mathbb{A}_0^{-1} \mathbb{E}_{0e} w_{0,e} \end{aligned} \quad (2.38)$$

Then, we can define $w = \tilde{R}_1^{0T} R_{1,e}^p R_{0,e}^{pT} w_{0,e} + \tilde{R}_0^{0T} R_0^p R_{1,e}^{pT} w_{1,e}$ such that:

$$\begin{aligned} R_{0,e}^p w &= w_{1,e} \quad \text{and} \quad R_{1,e}^p w = w_{0,e} \\ R_{0,e}^p P_g w &= \lambda R_{0,e}^p w \\ R_{1,e}^p P_g w &= \lambda R_{1,e}^p w \end{aligned}$$

As w has non-nul components only on the components belonging to Γ , we conclude that:

$$\begin{aligned} P_g w &= \lambda w, \lambda \in \Lambda(P) \Rightarrow \lambda \in \Lambda(P_g) \\ \Lambda(P_g) &= \Lambda(P) \end{aligned}$$

□

Remark 6. *It is therefore possible to establish the divergence of the method using the spectral radius of P or P_g . As long as the method does not stagnate, and even if the method diverges, we can obtain the converged solution thanks to the formula Eq. (2.27). The indication of the convergence of the method can be useful if one has to recalculate the error operator before recalculating the converged solution, indeed if the method diverges, one will have any interest to use in the equation Eq. (2.27) the first calculated iterations, whereas if the method converges it will be preferable to use the last ones.*

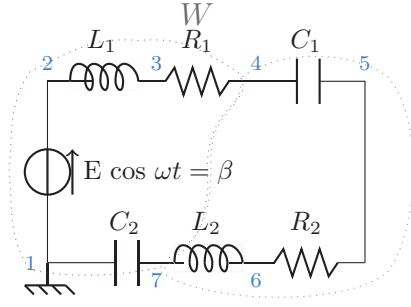
Several different strategies to compute P or P_g can be adopted depending on the problem to be solved. Indeed if the domain is small, then one can compute the global error operator P_g on the whole domain and accelerate the convergence of all the values of a blow.

On the other hand if the field to be solved is more consequent, to compute the error operator on the whole domain can require too many iterations of Schwarz if we compute the operator P_g numerically as it will be detailed in the next chapter. In this case, it is preferable to accelerate only the solution at the interfaces (the $W_{i,e}$) with the operator P and then to calculate the converged solution on the whole domain by carrying out a local resolution on each subdomain with the converged interface values.

In the next section we illustrate the numerical result on the RAS applied to two partitions with an homogeneous modeling that can be EMT or TS modeling.

2.5 Numerical results

We consider a linear RLC circuit of Figure 2.1. This is a single-loop circuit, so the different sub-domains will necessarily be strongly coupled. It contains the basic components used in the electrical field to model most phenomena. The DAE associated with the small circuit is written component by component. We do not make any simplification usually done when there are several resistors or inductors in series, to keep more equations. Instead of considering the single loop current, we will consider that there is one current per component, which will allow us to observe more phenomena like error propagation. In order to keep the right number of equations and variables, and to be able to put two inductors in series, we add a degree of freedom (that is, we remove one of the current equality equations).



$$\begin{aligned}
 v_1 &= 0 \\
 v_2 - v_1 - E - Z_s i_{12} &= 0 \\
 v_3 - v_2 - L_1 \frac{di_{23}}{dt} &= 0 \\
 v_4 - v_3 - R_1 i_{34} &= 0
 \end{aligned}$$

$$\begin{aligned}
 C_1 \left(\frac{dv_5}{dt} - \frac{dv_4}{dt} \right) - i_{45} &= 0 \\
 v_6 - v_5 - R_2 i_{56} &= 0 \\
 v_7 - v_6 - L_2 \frac{di_{67}}{dt} &= 0 \\
 C_2 \left(\frac{dv_1}{dt} - \frac{dv_7}{dt} \right) - i_{71} &= 0 \\
 i_{12} - i_{23} &= 0 \\
 i_{23} - i_{34} &= 0 \\
 i_{34} - i_{45} &= 0 \\
 i_{45} - i_{56} &= 0 \\
 i_{56} - i_{67} &= 0 \\
 i_{67} - i_{71} &= 0
 \end{aligned}$$

Figure 2.1: Linear RLC circuit and its associated EMT modeling DAE system with

$x = \{v_1, i_{23}, v_4, v_5, i_{67}, v_7\}$ and $y = \{v_2, i_{12}, v_3, i_{34}, i_{45}, i_{56}, v_6, i_{71}\}$. $L1 = L2 = 0.7$, $C1 = C2 = 1.10^{-6}$, $R1 = R2 = 77$, $Z_s = 1.10^{-6}$, $\omega_0 = 2\pi 50$.

Figure 2.2 is an example of this cutting for a small RLC circuit. The small linear system associated with the RLC circuit is partitioned into two subdomains using non-overlapping graph partitioning (Figure 2.2 on top) and with an overlap of 6 components (Figure 2.2 bottom). This use of graph partitioning aims to have an equivalent computational load for each subdomain in a homogeneous case (the two subdomains are modeled in the same EMT or TS way).

As this decomposition is graph-based, it does not prejudice the convergence or divergence of the resulting Schwarz method. In this example the domain W is cut into two non-overlapping partitions $W_1^0 = \{v_1, v_2, i_{12}, i_{71}, v_3, v_7, i_{23}, i_{34}\}$ and $W_2^0 = \{v_4, i_{67}, v_5, v_6, i_{56}\}$, or two overlapping partitions $W_1^1 = \{v_1, v_2, i_{12}, i_{71}, v_3, v_7, i_{23}, i_{34}, i_{71}, v_4, i_{67}\}$ and $W_2^1 = \{v_3, v_7, i_{34}, v_4, i_{67}, v_5, v_6, i_{56}\}$ obtained with including the unknowns not belonging to W_i^0 that correspond to the data-dependencies in the equations involving the unknowns of W_i^0 .

There are two interesting things to notice about this distribution:

The first is that the equation used to calculate v_7 is calculated in the equations of the domain W_1 , this equation is $L_2 \frac{di_{67}}{dt} + v_6 = v_7$, which then is rewritten after a Backard Euler discretization as: $L_2 \frac{i_{67}^{n+1} - i_{67}^n}{\Delta t} + v_6^{n+1} = v_7^{n+1}$. However, i_{67}^{n+1} and i_{67}^n are values computed by the equation of the second domain W_2 , we see in this case the impact of the choice of the division before or after discretization. The second point to note here is that the physical components (the inductor here for example)

can be partitioned between several subdomains with this kind of division, which can cause problems in the use of certain tools, as we will see in chapter 5.

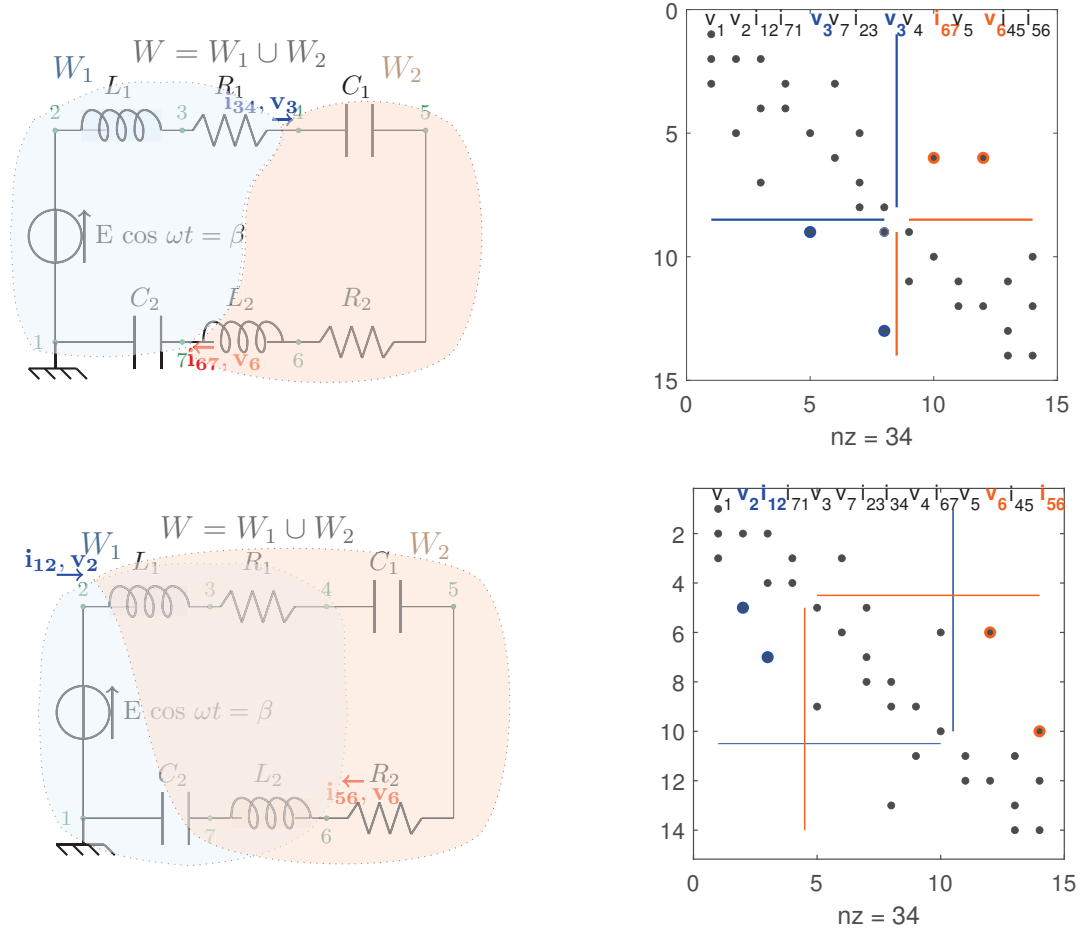


Figure 2.2: Graph partitioning of the RLC circuit in two subdomains and the associated matrix partitioning without overlap (top) and with overlap of 1 (bottom). EMT case

Figure 2.2 represents, on the left the cut circuit with and without recovery, and on the right their associated state space matrix \mathbb{A} , with only the representation of the load (there is a point on i, j if $\mathbb{A}(i, j) \neq 0$) these state space matrices are split into two parts corresponding to the two sub-domains. We can observe that in both cases (with and without overlap) each sub-matrix needs two values calculated by the other part (these values are represented by circled points).

Let us detail the build of the matrix \mathbb{A} : the backward Euler discretizing of the RLC circuit DAE system leads to the following discrete equations:

$$\begin{aligned}
v_1^{n+1} &= 0 \\
v_2^{n+1} - v_1^{n+1} - E - Z_s i_{12}^{n+1} &= 0 & i_{12}^{n+1} - i_{23}^{n+1} &= 0 \\
\Delta t v_3^{n+1} - \Delta t v_2^{n+1} - L_1 i_{23}^{n+1} &= -L_1 i_{23}^n & i_{23}^{n+1} - i_{34}^{n+1} &= 0 \\
v_4^{n+1} - v_3^{n+1} - R_1 i_{34}^{n+1} &= 0 & i_{34}^{n+1} - i_{45}^{n+1} &= 0 \\
C_1 v_5^{n+1} - C_1 v_4^{n+1} - \Delta t i_{45}^{n+1} &= C_1 v_5^n - C_1 v_4^n & i_{45}^{n+1} - i_{56}^{n+1} &= 0 \\
v_6^{n+1} - v_5^{n+1} - R_2 i_{56}^{n+1} &= 0 & i_{56}^{n+1} - i_{67}^{n+1} &= 0 \\
\Delta t v_7^{n+1} - \Delta t v_6^{n+1} - L_2 i_{67}^{n+1} &= -L_2 i_{67}^n & i_{71}^{n+1} - i_{12}^{n+1} &= 0 \\
C_2 v_1^{n+1} - C_2 v_7^{n+1} - i_{71}^{n+1} \Delta t &= C_2 v_1^n - C_2 v_7^n
\end{aligned}$$

The time integration from t_n to t_{n+1} thus requires to solve a linear system $\mathbb{A}z^{n+1} = b$. The partitioning of the graph of the matrix \mathbb{A} obtained by replacing its non-zero coefficients by 1's, allows to separate the system of discrete equations into two subsystems of equations and provides the partitioning of the unknowns into two partitions: $W_1^0 = \{v_1, v_2, i_{12}, i_{71}, v_3, v_7, i_{23}, i_{34}\}$ and $W_2^0 = \{v_4, i_{67}, v_5, v_6, i_{56}\}$. The unknowns involved in a subsystem of equations but not in the partition of this subsystem are the inputs of this subsystem.

$$\begin{aligned}
v_1^{n+1} &= 0 \\
v_2^{n+1} - v_1^{n+1} - E - Z_s i_{12}^{n+1} &= 0 & v_4^{n+1} &= v_3^{n+1} + R_1 i_{34}^{n+1} \\
\Delta t v_3^{n+1} - \Delta t v_2^{n+1} - L_1 i_{23}^{n+1} &= -L_1 i_{23}^n & C_1 v_5^{n+1} - C_1 v_4^{n+1} - \Delta t i_{45}^{n+1} &= C_1 v_5^n - C_1 v_4^n \\
\Delta t v_7^{n+1} &= -L_2 i_{67}^n + \Delta t v_6^{n+1} + L_2 i_{67}^{n+1} & v_6^{n+1} - v_5^{n+1} - R_2 i_{56}^{n+1} &= 0 \\
C_2 v_1^{n+1} - C_2 v_7^{n+1} - i_{71}^{n+1} \Delta t &= C_2 v_1^n - C_2 v_7^n & -i_{45}^{n+1} &= -i_{34}^{n+1} \\
i_{12}^{n+1} - i_{23}^{n+1} &= 0 & i_{45}^{n+1} - i_{56}^{n+1} &= 0 \\
i_{23}^{n+1} - i_{34}^{n+1} &= 0 & i_{56}^{n+1} - i_{67}^{n+1} &= 0 \\
i_{71}^{n+1} - i_{12}^{n+1} &= 0
\end{aligned}$$

In the EMT case, each subdomain needs two values from the other to solve its equations. In the TS case, as we choose to solve harmonics 0 and 1 and to solve real and imaginary part apart (the imaginary part of the harmonic 0 is always 0), each subdomain needs six values from the other to solve its own equations.

Figures 2.3 and 2.4 represent the linear divergence or convergence of RAS for EMT-EMT and TS-TS co-simulation. In the case RAS it is necessary to take an error out of two to see the linear convergence (and all in the case RMS). It can be observed that the size of the time step greatly influences the convergence of the method. Here the method diverges more slowly for the EMT and TS by taking a large time step, which seems counter-intuitive. These results are supported by the calculation of the error operator spectral radius value in Table 2.2.

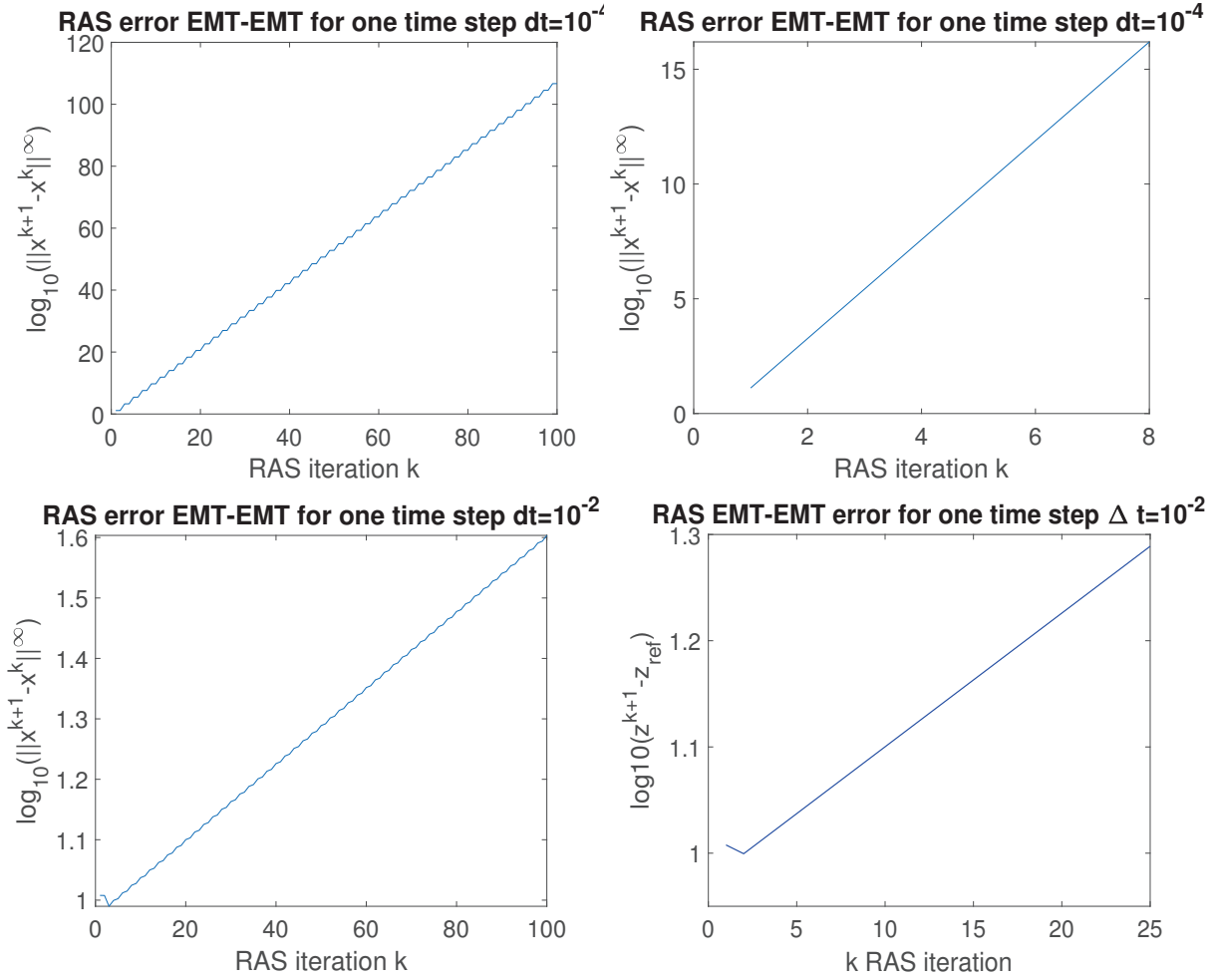


Figure 2.3: Error between two consecutive RAS iterations for EMT-EMT case with top $\Delta t_{emt} = 10^{-4}$, bottom $\Delta t_{emt} = 10^{-2}$. By taking all the errors for the two on the right and taking only one out of two for the two on the left.

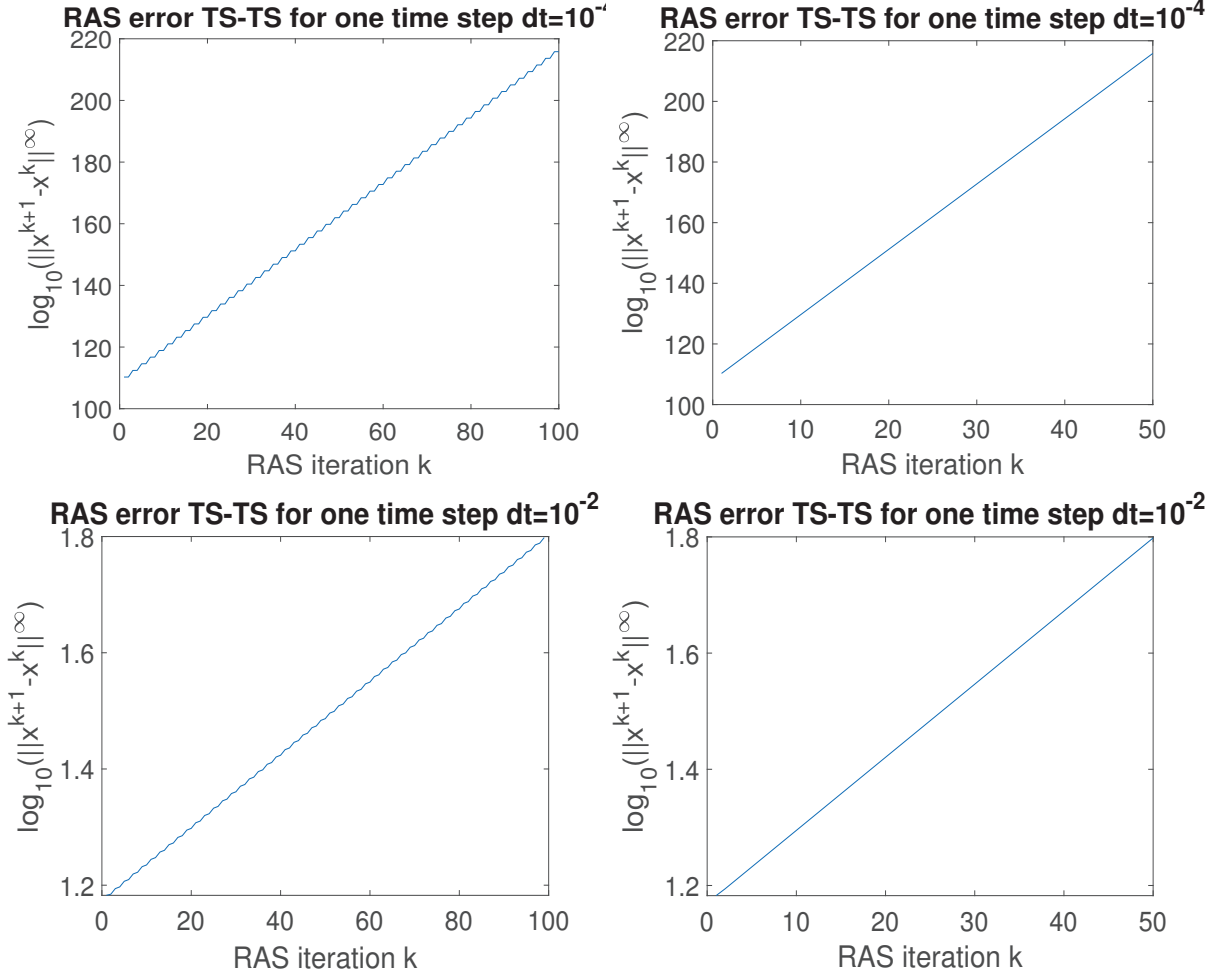


Figure 2.4: Error between two consecutive RAS iterations for TS-TS case with top $\Delta t_{emt} = 10^{-4}$, bottom $\Delta t_{emt} = 10^{-4}$. By taking all the errors for the two on the right and taking only one out of two for the two on the left.

Table 2.2 gives the larger eigenvalue in modulus for the P RAS error operator for the EMT modeling and for the TS modeling main harmonic $k = 1$ applied to the RLC circuit. In both cases EMT and TS modeling the eigenvalue modulus is greater than one, so the method diverges.

We can observe that, contrary to what one might expect, the overlap does not impact the divergence of the method.

The time step increasing from $\Delta t = 2 \cdot 10^{-4}$ to $\Delta t = 2 \cdot 10^{-2}$ has a beneficial effect on the TS-TS DDM divergence. Nevertheless, the divergence is purely linear and the Aitken's acceleration (2.27) can be performed after the first iteration for each time step.

$\lambda(P)$	without overlap	with overlap	Schwarz	Δt
EMT	$\pm 11.94i$	$\pm 11.94i$	RAS	2.10^{-4}
EMT	$\pm 1.0146i$	$\pm 1.0146i$	RAS	2.10^{-2}
TS $k=0$	$0.3704 \pm 11.939i$	$0.3704 \pm 11.939i$	RAS	2.10^{-4}
TS $k=1$	$\pm 11.94i$	$\pm 11.94i$	RAS	2.10^{-4}
TS $k=0$	$0.0720 \pm 0.9468i$	$0.0720 \pm 0.9468i$	RAS	2.10^{-2}
TS $k=1$	$\pm 1.0146i$	$\pm 1.0146i$	RAS	2.10^{-2}

Table 2.2: Larger eigenvalue for P error operator for RAS and EMT modeling ($\Delta t_{emt} = 2.10^{-4}$, $\Delta t_{emt} = 2.10^{-3}$), and for RAS and TS $k = 0, 1$ ($\Delta t_{ts} = 2.10^{-4}$, $\Delta t_{ts} = 2.10^{-3}$) modeling.

Figure 2.5 and Figure 2.6 show the results of accelerated RAS with the Aitken acceleration technique, compared in each case with the monolithic reference. Co-simulation gives results similar to monolithic. As we can observe on 2.5, as long as the time step is small enough, the simulation of the EMT and the dynamic phasor gives really similar levels of precision, and, at equal time steps, the computational load of dynamic phasors is heavier than that of EMT. The advantage of the dynamic phasor is that we can take larger time steps and as can be seen in figure 2.6 by taking a larger time step, the EMT cannot give realistic results, whereas the dynamic phasor can even if the level of detail is reduced.

2.6 Conclusion

In this section, we presented the Restricted Additive Schwarz method in its general application. We have adapted the method to the DAE system discretized in time. We have shown that the linear convergence property of the RAS still remains for the discrete DAE system. The way we will cut the DAE system for this thesis work has been introduced. The numerical results showed that the same behavior cannot be expected for the RAS method applied to linear circuit DAE systems as for the spatial division of PDEs with elliptical term systems. In particular, the size of the overlap does not improve convergence. In particular, whatever the homogeneous modeling of the partitions, the RAS turns out to be a divergent method. Nevertheless, as this divergence remains linear, we derive an efficient method of RAS acceleration. This method makes it possible to accelerate the calculated solution towards the true solution thanks to the technique of acceleration of the convergence of Aitken. One also benefits from an integration with constant time step, which makes it possible by calculating the operator of error associated with the first step of time and by reusing it for the following time steps, to carry out only one iteration RAS by time step.

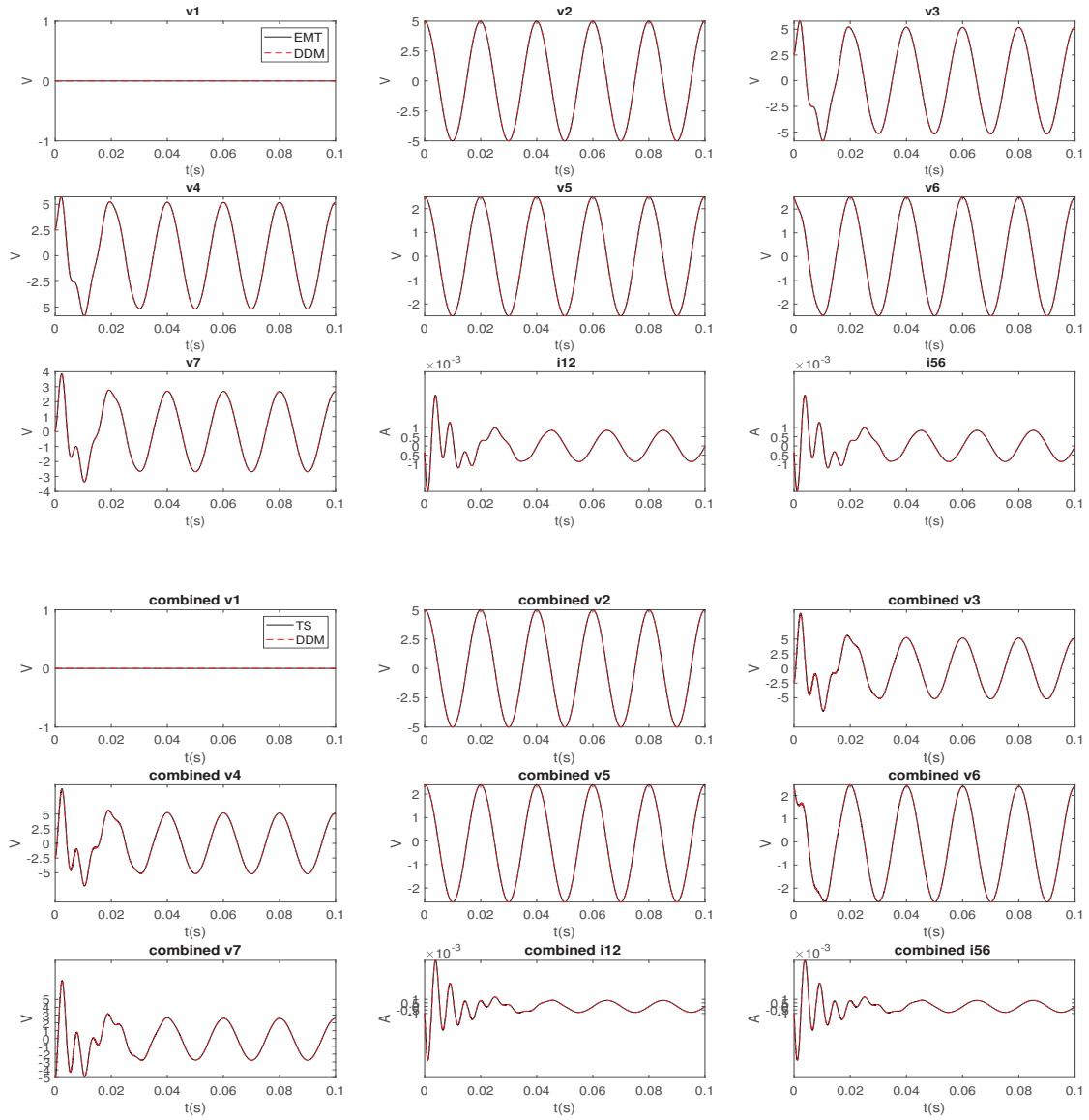


Figure 2.5: Homogeneous DDM results comparison with DAE monodomain: (Top) RAS for EMT modeling with $\Delta t_{emt} = 1.10^{-4}$ and (Bottom) RAS for TS modeling with $\Delta t_{ts} = 2.10^{-4}$.

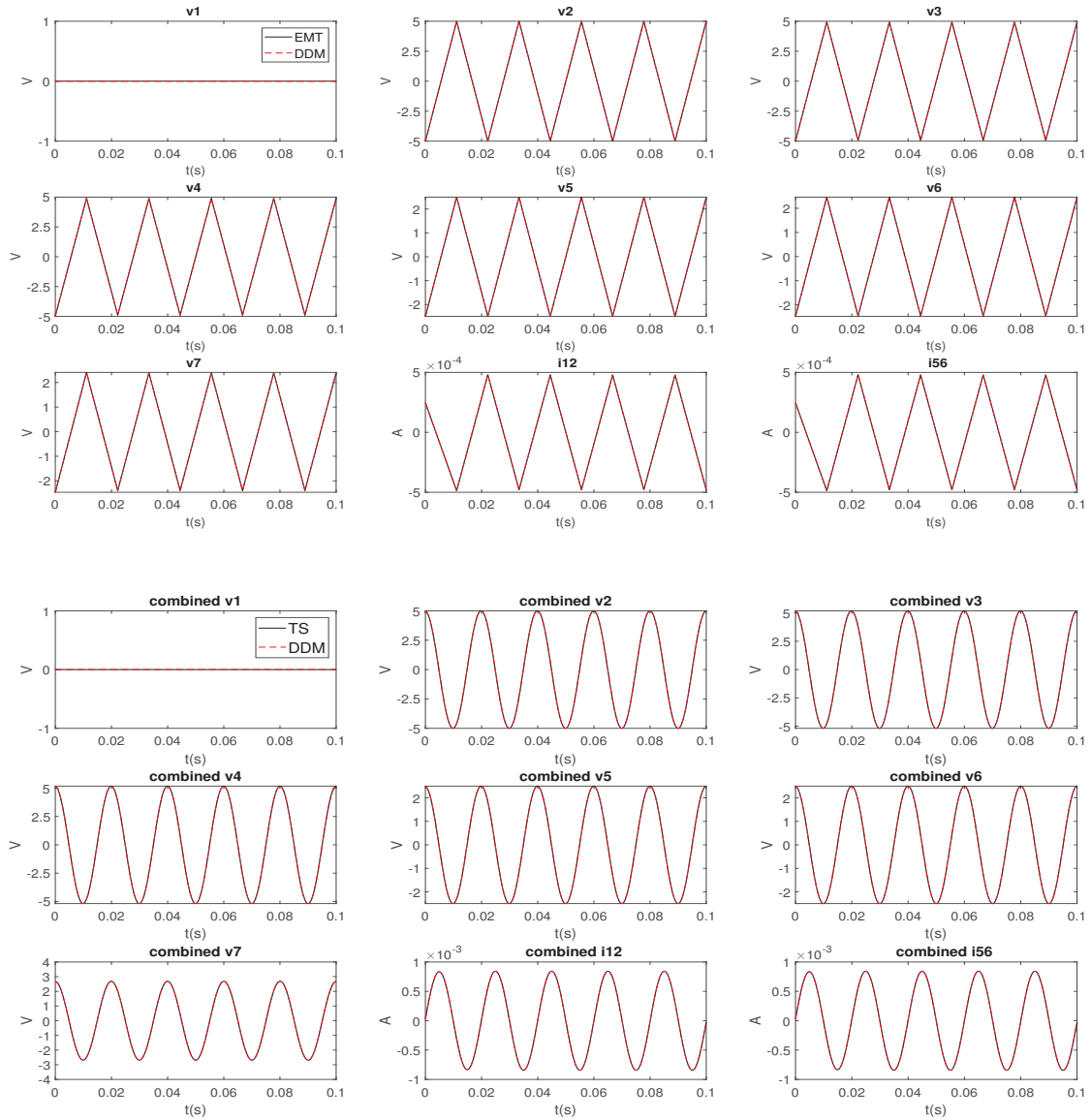


Figure 2.6: Homogeneous DDM results comparison with DAE monodomain: (Top) RAS for EMT modeling with $\Delta t_{emt} = 1.10^{-2}$ and (Bottom) RAS for TS modeling with $\Delta t_{ts} = 2.10^{-2}$.

Chapter 3

Schwarz method in the heterogeneous EMT-TS case

Contents

3.1	Introduction	45
3.2	Schwarz method for heterogeneous EMT-TS	47
3.2.1	TS side	47
3.2.2	EMT side	52
3.2.3	Initialization	56
3.2.4	Heterogeneous EMT-TS RAS formulation	57
3.3	Heterogeneous EMT-TS RAS error operator	57
3.3.1	Acceleration strategy	58
3.3.2	Error: a multifactorial resultant	59
3.4	Heterogeneous EMT-TS Numerical Results	61
3.4.1	Heterogeneous EMT-TS RAS convergence results	61
3.4.2	Qualitative results on the heterogeneous EMT-TS RAS	75
3.5	Conclusions	77

3.1 Introduction

In this chapter, we are going to set up the method that we are going to use to simulate a network with a part in EMT and a part in TS. Indeed, the goal is to

take advantage of both the accuracy of the EMT and the computational efficiency of the TS. We will adapt the Schwarz-Aitken method introduced in chapter 2 to the heterogeneous case. To co-simulate EMT-TS, additional difficulties related to the presence of two different types of modeling must be managed.

- The first difficulty to take into account is the transfer of information, more precisely the translation of values from one type of modeling to another. This translation must be as faithful as possible to the starting value in order to convey the information with great precision, it must be able to adapt to macro time steps and finally it must not call into question the pure linear convergence of the method, so the convergence can be accelerated using the Aitken's method.
- A second difficulty would be the difference in time step duration between TS and EMT, this must be taken into account for the exchange protocol. Most often it is proposed to choose time steps Δt_{ts} and Δt_{emt} such that $\Delta t_{ts} = m\Delta t_{emt}$ with $m \in \mathbb{N}^*$ and taking as macro time step the time step of the TS part (Δt_{ts}). This approach limits the fact of having variable time steps, however this is what we are going to do for the moment to facilitate the implementation of the method.
- Finally, the choice of partitioning can raise questions. Thanks to Aitken acceleration of the convergence technique, one is freed from the condition of partitioning which makes it possible the method to converge. However, the size of the EMT part must be a good compromise between detail and speed. It is also better to have subdomains with the same computational weight in order to get the most out of parallelization. This difficulty to be taken into account for EMT-TS co-simulation will also be dealt with in another chapter.

In this chapter we will focus on the adaptation of the Aitken-Schwarz method to a heterogeneous EMT-TS case, only the first difficulty is really essential to solve for this adaptation. First, we will rewrite the systems of equations associated with each subdomain with boundary conditions coming from another type of subdomain. First, in the case of a TS subdomain, we will introduce the EMT to TS translation operator. Next, we will apply the same approach in the case of an EMT subdomain and also introduce the translation operator from TS to EMT. Afterwards, we will talk about initialization, and finally, all this work will be assembled in order to obtain a heterogeneous Schwarz method.

In a second time, we will present a method making it possible to numerically calculate the operator of error. Subsequently, the impact of the heterogeneity of the

method on the computation of the error operator and on the acceleration strategy of Aitken will be studied. Specifically, the impact of translation operators on error will be discussed.

This chapter will be illustrated with numerical results. For this we will use the RLC circuit from chapter 2. Finally we will conclude.

3.2 Schwarz method for heterogeneous EMT-TS

In order to build the co-simulation, we will first take the representation of the discrete state space back and rewrite it for each of the representations in order to fix the notations. We choose to separate the differential variables (which we will denote x) from the purely algebraic variables (which we will denote y), assuming that $\Delta t_{ts} = m\Delta t_{emt}$. We will assume that we have the TS representation of a domain W as well as the EMT representation of the same domain. In simulations, it will not always be possible to have both TS and EMT representations on common parts of the network, but when it is possible, that makes it possible to define an overlap and thus to compare the solutions obtained with the two solvers. We will split as in 2.3.2, we get subdomains W_i . Let us rewrite the systems on a W_i subdomain assuming that the values at the artificial interfaces retrieved by an EMT subdomain (respectively TS) necessarily come from a TS subdomain (respectively EMT).

3.2.1 TS side

We take back the discret state space system 2.20, we need to add the boundary conditions.

For an W_i TS subdomain integrated from T^N to T^{N+1} :

$$\begin{aligned}
 & \underbrace{\begin{pmatrix} \mathbb{I}_{i_{ts}} - \Delta t_{i_{ts}} A_{i_{ts}} & B_{i_{ts}} \\ C_{i_{ts}} & D_{i_{ts}} \end{pmatrix}}_{\mathbb{A}_{i_{ts}}} \underbrace{\begin{pmatrix} x_{i_{ts}}^{N+1} \\ y_{i_{ts}}^{N+1} \end{pmatrix}}_{w_{i_{ts}}^{N+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_{i_{ts}} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,i_{ts}}} \underbrace{\begin{pmatrix} x_{i_{ts}}^N \\ y_{i_{ts}}^N \end{pmatrix}}_{w_{i_{ts}}^N} \\
 & + \underbrace{\begin{pmatrix} \mathbb{I}_{i,e_{ts}} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,i_{e_{ts}}}} \mathbb{T}_{ts}^{emt} \underbrace{\begin{pmatrix} X_{i,e_{emt}}^N \\ Y_{i,e_{emt}}^N \end{pmatrix}}_{Z_{i,e_{emt}}^N} + \underbrace{\begin{pmatrix} E_{i_{ts}}^A & E_{i_{ts}}^B \\ E_{i_{ts}}^C & E_{i_{ts}}^D \end{pmatrix}}_{\mathbb{E}_{i_{ts}}^{emt}} \underbrace{\begin{pmatrix} X_{i,e_{emt}}^{N+1} \\ Y_{i,e_{emt}}^{N+1} \end{pmatrix}}_{Z_{i,e_{emt}}^{N+1}} + G_{i_{ts}}^{N+1}. \quad (3.1)
 \end{aligned}$$

We take back all notations from the homogeneous case (cf 2.3.2). In order to introduce the operators used in (3.1) we need to introduce additional notations specific to the heterogeneous case. We recall that a signal $s(x)$ is said to be periodic

if there is a constant $T > 0$, known as the period, for which:

$$s(x + T) = s(x), \forall x \in D(s) \quad (3.2)$$

The frequency of this signal s is then given by: $f = \frac{1}{T}$, in electrical networks this frequency is normally 50Hz or 60Hz. And the pulsation is given by $\omega = 2\pi f$, the pulsation is expressed in radians/second.

We will call here the original pulsation (the one without frequency disturbance) of our signals ω_0 and the the associated period T_0 , this period is expressed in seconds. First denote $\tilde{m} = \frac{T_0}{\Delta t_{emt}}$ the number of emt time steps performed during a period and let K be the number of harmonics we choose to keep for the dynamic phasor modeling.

$Z_{i,e_{emt}} = R_{i,e}^p Z_{emt}$, we note n_{ie} the size of the dependencies of domain W_i with others domain, $Z_{i,e_{emt}} \in \mathbb{R}^{(\tilde{m} \times ie) \times 1}$.

As the representations are not the same in EMT and in TS, the boundary conditions resulting from an EMT subdomain must be transformed so as to be of the TS form. For this, we define the translation operator from EMT to TS :

$\mathbb{T}_{ts}^{emt} : \mathbb{R}^{(\tilde{m} \times ie) \times 1} \mapsto \mathbb{R}^{(2K \times ie) \times 1}$. More details on this transformation are explained later in this chapter (see section 3.2.1.1).

We define the operator $\mathbb{E}_{ie}^{emt} = \underbrace{R_i^p \mathbb{A}_{emt} R_{i,c}^{pT}}_{\mathbb{E}_{ie}} \mathbb{T}_{ts}^{emt}$, with \mathbb{E}_{ie} defined the same way as

in the homogeneous case. As in the homogeneous case, $\mathbb{I}_{d,ie_{ts}} \mathbb{T}_{ts}^{emt} Z_{i,e_{emt}}^N$ is due to the fact that differential terms on the interface unknowns may be involved due to the splitting.

$Z_{i,e_{emt}}^N$ is a much larger vector than $\mathbb{T}_{ts}^{emt} Z_{i,e_{emt}}^N$. Moreover the $Z_{i,e_{emt}}^N$ has already been transformed into a TS form at the previous time step. Rather than keeping a large vector and recomputing the translation, we will keep $\mathbb{T}_{ts}^{emt} Z_{i,e_{emt}}^N$ calculated during the previous time step. Therefore, one can gather all the terms of the preceding time step in the same vector: $w_{i,ie}^N = w_i^N + \mathbb{T}_{ts}^{emt} Z_{ie_{emt}}^N$, we rewrite the equation (3.1).

$$\begin{aligned} & \underbrace{\begin{pmatrix} \mathbb{I}_{i_{ts}} - \Delta t_{i_{ts}} A_{i_{ts}} & B_{i_{ts}} \\ C_{i_{ts}} & D_{i_{ts}} \end{pmatrix}}_{\mathbb{A}_{i_{ts}}} \underbrace{\begin{pmatrix} x_{i_{ts}}^{N+1} \\ y_{i_{ts}}^{N+1} \end{pmatrix}}_{w_{i_{ts}}^{N+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_{i,ie_{ts}} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,i,ie_{ts}}} \underbrace{\begin{pmatrix} x_{i,ie_{ts}}^N \\ y_{i,ie_{ts}}^N \end{pmatrix}}_{w_{i,ie_{ts}}^N} \\ & + \underbrace{\begin{pmatrix} E_{i_{ts}}^A & E_{i_{ts}}^B \\ E_{i_{ts}}^C & E_{i_{ts}}^D \end{pmatrix}}_{\mathbb{E}_{ie_{ts}}^{emt}} \underbrace{\begin{pmatrix} X_{i,e_{emt}}^{N+1} \\ Y_{i,e_{emt}}^{N+1} \end{pmatrix}}_{Z_{i,e_{emt}}^{N+1}} + G_{i_{ts}}^{N+1}. \end{aligned} \quad (3.3)$$

Definition 5. *The $p + 1$ iteration of the Restrictive Additive Schwarz algorithm in the discrete case is written locally for the $W_{i,ts}^p$ partition from the TS type and with integrating between T^N and T^{N+1} , if $\mathbb{A}_{i,ts}$ invertible, as:*

$$w_{i,ts}^{N+1,p+1} = \mathbb{A}_{i,ts}^{-1} (\mathbb{I}_{i,ie_{ts}} w_{i,ie}^{N,\infty} - \mathbb{E}_{i,ts}^{emt} Z_{i,e_{emt}}^{N+1,p} + G_{i,ts}^{N+1}) \quad (3.4)$$

3.2.1.1 Translation from EMT to Dynamic Phasor

The three most common types of techniques used to translate EMT information into TS are:

- Curve fitting techniques as the least-squares curve fitting technique used by Plumier[94].
- Methods of changing referential frames such as the α, β method used by Konara [81] and by Zamroni [115], or as the direct-quadrature-zero transformation method (dq0)[94].
- Methods based on the Fast Fourier Transform (FFT), as performed by Kumara in[82].

Each of these techniques has advantages and disadvantages, for the curve fitting technique it is the least precise of the 3 techniques, moreover it is not necessarily a linear operator (although often it is linear because it is often a linear interpolation). The methods of changing referential frame has the merit of being instantaneous, on the other hand it assumes that the three-phase current is necessarily balanced, which already induces a loss of information, moreover we cannot hope to recover the slightest information on what has happened during the EMT intermediate time steps. This way of doing things will therefore be the best if what happens in the EMT part does not have too strong an impact on the TS part. The FFT is the most precise of the 3 transformations, and is linear, its disadvantage being that it is necessary to wait for a period to be able to apply it, which strongly restricts the choice of macro time steps.

The two methods which are going to be used in this work are the DQ0 transformation and, in this chapter, an FFT-based method, slightly modified in order to get rid of the period restriction. We will therefore provide more details on these methods.

- DQ0-transformation: It is an instantaneous transformation that conserves power. Consider a balanced three-phase signal $s_{emt} = (s_a, s_b, s_c)^t$, we obtain

after transformation DQ0 $s_{ts} = (\tilde{s}_{re}, \tilde{s}_{im}, \tilde{s}_0)^t$.

The transformation takes the form: $s_{ts} = M_{dq0} s_{emt}$

$$\text{with } M_{dq0} = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\omega_0) & \cos(\omega_0 - \frac{2\pi}{3}) & \cos(\omega_0 + \frac{2\pi}{3}) \\ -\sin(\omega_0) & -\sin(\omega_0 - \frac{2\pi}{3}) & -\sin(\omega_0 + \frac{2\pi}{3}) \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

The inverse transformation is given by $s_{emt} = M_{dq0}^{-1} s_{ts}$

$$\text{with } M_{dq0}^{-1} = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\omega_0) & -\sin(\omega_0) & \frac{\sqrt{2}}{2} \\ \cos(\omega_0 - \frac{2\pi}{3}) & -\sin(\omega_0 - \frac{2\pi}{3}) & \frac{\sqrt{2}}{2} \\ \cos(\omega_0 + \frac{2\pi}{3}) & -\sin(\omega_0 + \frac{2\pi}{3}) & \frac{\sqrt{2}}{2} \end{bmatrix}$$

Remark 7. *The DQ0 transform must be applied for each harmonic that has been chosen to be kept for the TS. The above matrix is used to obtain the fundamental harmonic.*

- The FFT: It is an algorithm which makes it possible to obtain the Discrete Fourier Transform (DFT) of a signal in an inexpensive way. The Fourier transform makes it possible, thanks to a spectral decomposition, to decompose a periodic signal into a sequence of harmonics.

The Fourier transform of a signal s is given by $\tilde{S}(k) = \int_{-\infty}^{\infty} s(\tau) e^{jk\omega_0\tau} d\tau$. If s is periodic with period T then the Fourier transform can be related to an observation window $\tau \in [t - T, t]$: $\tilde{S}(k) = \frac{1}{T} \int_{t-T}^t s(\tau) e^{-jk\omega_0\tau} d\tau$.

The DFT of a signal s is an approximation of the Fourier transform of this signal, it's defined as $S(k) = \frac{1}{N} \sum_{n=0}^{N-1} s(n) e^{-2i\pi k \frac{n}{N}}$ with $0 < k \leq N$. The sampling N must cover a period T of s .

To translate from the EMT to the dynamic phasor, we apply a fast Fourier transform on a history of the size of a period, T_0 , containing the values taken by the part calculated by the EMT at the intermediate time steps. We can then recover all the modes corresponding to the frequencies that we have chosen to keep in the dynamic phasor representation.

As previously explained, to perform an FFT on a signal, the history must cover a period of this signal. We can therefore take a macro time step of size T_0 , the period and fill the history with the $m = \tilde{m}$ intermediate time steps. Since we decided that the macro time steps would be the TS time steps, we take $\Delta t_{ts} = T_0$ the size of a Period.

However, it is preferable to be able to take smaller TS time steps. We therefore take a Δt_{ts} shorter than one period, however to perform the FFT, a history of

the size of a period is always necessary, a “sliding” history will be produced. To do this, we will delete the m (reminder $\Delta t_{ts} = m\Delta t_{emt}$) oldest values from the beginning of the history and put the m new values at the end of the history. This operation is repeated at each instant of passage of information, i.e. after each TS time step (and therefore after m time steps EMT).

Definition 6. We recall that $m = \frac{\Delta t_{ts}}{\Delta t_{emt}}$ and $\tilde{m} = \frac{T_0}{\Delta t_{emt}}$, let's also denote $M = \frac{\tilde{m}}{m}$ the number of Δt_{ts} during a period. The history of the integration of t^N to t^{N+1} , for the p^{th} Schwarz iteration $Z_{ie_{emt}}^{N+1,p}$ is defined as follows:

$$Z_{ie_{emt}}^{N+1,p} = \underbrace{[z_{ie_{emt}}^{m(N-M+1)+1,\infty}, \dots, z_{ie_{emt}}^{m(N-M+1)+m,\infty}, \dots, z_{ie_{emt}}^{m \times N,\infty}]_{Z_{ie_{emt}}^p}}_{\text{Unactiv boundary conditions}}, \underbrace{[z_{ie_{emt}}^{m \times N+1,p}, \dots, z_{ie_{emt}}^{m \times (N+1),p}]}_{\text{Activ Boundary conditions}}$$

However, an FFT on a moving story is also moving. See Figure 3.1 where we compare the harmonic 1st of a cos signal calculated with a dynamic phasor and with an FFT (applied to the EMT signal), we can observe, on the top Figure, that the FFT and the dynamic phasor meet only once per full period.

Indeed, for the equation (3.2.1.1) the observation window will no longer be $[t-T_0, t]$ but $[t-T_0 + \alpha\Delta t_{ts}, t + \alpha\Delta t_{ts}]$, applying the appropriate change of variable in the Fourier transform formula, one can observe a $e^{jkw_0\alpha\Delta t_{ts}}$ gap: Indeed lets take back the 3.2.1.1 formula on the $[t-T_0 + \alpha\Delta t_{ts}, t + \alpha\Delta t_{ts}]$ observation window :

$$\tilde{S}(k) = \frac{1}{T} \int_{t-T_0+\alpha\Delta t_{ts}}^{t+\alpha\Delta t_{ts}} s(\tau) e^{jkw_0\tau} d\tau, \text{ by setting } X = \tau - \alpha\Delta t_{ts}t$$

$$\tilde{S}(k) = \frac{1}{T} \int_{t-T_0}^t s(X) e^{jkw_0(X+\alpha\Delta t_{ts})} dX$$

There is therefore a gap that appears:

$$\tilde{S}(k) = e^{jkw_0\alpha\Delta t_{ts}} \frac{1}{T} \int_{t-T_0}^t s(X) e^{jkw_0X} dX, \text{ that we will compensate by applying a } e^{-jkw_0\alpha\Delta t_{ts}} \text{ gap to the FFT.}$$

See the bottom figure of Figure 3.1 where we again compare the harmonic 1st of a cos signal calculated with the a dynamic phasor and with an FFT, and apply the appropriate correction at the FFT, it can be observed that the FFT of the EMT signal correspond to the dynamic phasor.

Definition 7. The translation operator \mathbb{T}_{ts}^{emt} consists of:

- Apply an FFT with a selection of the K mods corresponding to the K selected harmonics.
- Next apply a compensation $e^{-jw_0\alpha\Delta t_{ts}}$.
- Finally separate the real and imaginary part of the obtained values.

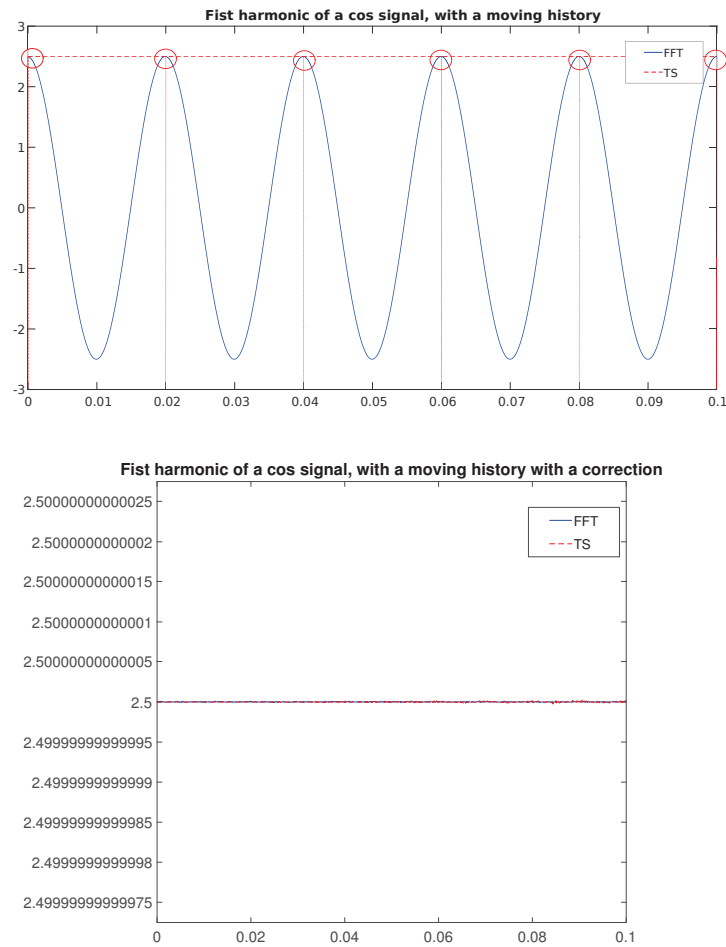


Figure 3.1: 1st Harmonic of a moving history, computed with PhasorDynamic and an FFT. Top: without applying any correction, Bottom: with applying a correction

Proposition 6. *The conditions imposed at the boundaries of a TS domain $W_{i,ts}^p$ can be considered as analogous to the Dirichlet conditions, in fact the differentiated part of the boundary conditions having passed into $w_{i,ie_{ts}}^{N, \infty}$ they are fixed on the iterations of Schwarz and have no impact on the conditions at the interfaces.*

3.2.2 EMT side

As for the TS sub-domains, for an EMT sub-domain, we use the discrete state space system Eq. (2.21), we have to adapt the data dependencies since the data comes from TS sub-domains.

For an $W_{i_{emt}}$ EMT subdomain integrated from t^n to $t^{n+1} = t^n + \Delta t_{emt}$ and $t^n, t^{n+1} \in [T^N, T^{N+1}]$ is written:

$$\begin{aligned} & \underbrace{\begin{pmatrix} \mathbb{I}_{i_{emt}} - \Delta t_{emt} A_{i_{emt}} & B_{i_{emt}} \\ C_{i_{emt}} & D_{i_{emt}} \end{pmatrix}}_{\mathbb{A}_{i_{emt}}} \underbrace{\begin{pmatrix} x_{i_{emt}}^{n+1} \\ y_{i_{emt}}^{n+1} \end{pmatrix}}_{w_{i_{emt}}^{n+1}} = \underbrace{\begin{pmatrix} \mathbb{I}_{i_{emt}} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{d,i_{emt}}} \underbrace{\begin{pmatrix} x_{i_{emt}}^n \\ y_{i_{emt}}^n \end{pmatrix}}_{w_{i_{emt}}^n} \\ + & \underbrace{\begin{pmatrix} E_{i_{emt}}^A & E_{i_{emt}}^B \\ E_{i_{emt}}^C & E_{i_{emt}}^D \end{pmatrix}}_{E_{i_{emt}}^{ts}} \mathbb{T}_{emt}^{ts}(t^{n+1}) \underbrace{\begin{pmatrix} x_{i,e_{ts}}^{N+1} \\ y_{i,e_{ts}}^{N+1} \end{pmatrix}}_{W_{i,e_{ts}}^{N+1}}^{N+1} + \underbrace{\begin{pmatrix} \mathbb{I}_{d,i_{emt}} & 0 \\ 0 & 0 \end{pmatrix}}_{\mathbb{I}_{i_{emt}}} \mathbb{T}_{emt}^{ts}(t^n) \underbrace{\begin{pmatrix} x_{i,e_{ts}}^{N+1} \\ y_{i,e_{ts}}^{N+1} \end{pmatrix}}_{W_{i,e_{ts}}^{N+1}} + G_{i_{emt}}^{n+1} \end{aligned} \quad (3.5)$$

The operators are similar to that of the homogeneous Schwarz domain decomposition method. We define the operator $\mathbb{I}_{d,i_{emt}} = R_{i_{emt}}^p \mathbb{I}_{d_{emt}} R_{i,e_{emt}}^{pT}$. As EMT and TS data do not have the same shape, we introduce a translation operator from TS to EMT at time t^q : $\mathbb{T}_{emt}^{ts}(t^q) : (2K \times ie) \times 1 \mapsto \mathbb{R}^{ie \times 1}$, (with K the number of harmonics that we choose to keep). More details on this transformation are explained later in this chapter (see section 3.2.2.1).

$\mathbb{I}_{i_{emt}} \mathbb{T}_{emt}^{ts}(t^n) W_{i,e_{ts}}^{N+1}$ is the part due to the fact that differential terms of the interface unknowns may be involved due to splitting.

In a subdomain modeled by EMT, as the time steps Δt_{emt} are shorter than the macro time steps, the part due to the discretization of the interface values must therefore be calculated at each EMT micro time steps, it is therefore necessary to add the term $\mathbb{I}_{i_{emt}} \mathbb{T}_{emt}^{ts}(t^n) W_{i,e_{ts}}^{N+1}$. Let us consider the m micro time steps realized

by the EMT and by considering $t^n = T^N$, we can rewrite the behavior of the EMT on the whole time step of T^N to T^{N+1} as:

$$\begin{aligned} & \underbrace{\begin{pmatrix} I & & & & & \\ -\mathbb{I}_{i_{emt}} & \mathbb{A}_{i_{emt}} & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & -\mathbb{I}_{i_{emt}} & \mathbb{A}_{i_{emt}} \\ & & & & & -\mathbb{I}_{i_{emt}} & \mathbb{A}_{i_{emt}} \end{pmatrix}}_{\mathbb{H}_{i_{emt}}} \underbrace{\begin{pmatrix} w_{i_{emt}}^n \\ w_{i_{emt}}^{n+1} \\ \vdots \\ w_{i_{emt}}^{n+m-1} \\ w_{i_{emt}}^{n+m} \\ w_{i_{emt}}^{n+m} \end{pmatrix}}_{\mathbb{W}_{i_{emt}}^{N+1}} = \\ & \underbrace{\begin{pmatrix} I & & & & & \\ -\mathbb{I}_{d,i_{emt}} & E_{i_{emt}}^{ts} & & & & \\ & & \ddots & & & \\ & & & -\mathbb{I}_{d,i_{emt}} & E_{i_{emt}}^{ts} & \\ & & & & -\mathbb{I}_{d,i_{emt}} & E_{i_{emt}}^{ts} \end{pmatrix}}_{\mathbb{E}_{i_{emt}}^{ts}} \underbrace{\begin{pmatrix} (x^n, y^n)^t \\ W_{i,e_{ts}}^{N+1}(t^{n+1}) \\ \vdots \\ W_{i,e_{ts}}^{N+1}(t^{n+m-1}) \\ W_{i,e_{ts}}^{N+1}(t^{n+m}) \end{pmatrix}}_{\mathbb{W}_{i,e_{ts}}^{N+1}} + \underbrace{\begin{pmatrix} 0 \\ G_{i_{emt}}^{n+1} \\ \vdots \\ G_{i_{emt}}^{n+m-1} \\ G_{i_{emt}}^{n+m} \\ G_{i_{emt}}^{n+m} \end{pmatrix}}_{\mathbb{G}_{i_{emt}}^{N+1}} \end{aligned} \quad (3.6)$$

With $W_{i,e_{ts}}^{N+1}(t^{n+1}) = \mathbb{T}_{emt}^{ts}(t^{n+1})W_{i,e_{ts}}^{N+1}$

Definition 8. The $p + 1$ iteration of the Restrictive Additive Schwarz algorithm in the discrete case is written locally for the $W_{i,ts}^p$ partition from the EMT type and with integrating between T^N and T^{N+1} , if $\mathbb{H}_{i_{emt}}$ invertible, as:

$$\mathbb{W}_{i_{emt}}^{N+1,p+1} = \mathbb{H}_{i_{emt}}^{-1} (\mathbb{E}_{i_{emt}}^{ts} \mathbb{W}_{i,e_{ts}}^{N+1,p} + \mathbb{G}_{i_{emt}}^{N+1}) \quad (3.7)$$

3.2.2.1 Translation from Dynamic Phasor to EMT

To rebuild an EMT signal from the coefficients of the dynamic phasor, there are two main possibilities:

- apply the inverse DQ0 transformation (on each harmonic), this method is explained in subsection 3.2.1.1.
- respect the way the dynamic phasor is built and recombine the harmonics.

In this thesis, when the translation from EMT to TS is the DQ0 transform, we will use the inverse DQ0 transform to translate from TS to EMT. In this chapter, we have chosen to use a Fourier transform to translate from EMT to TS, so to be consistent, we will use harmonic recombination to translate from TS to EMT. We therefore choose to apply the (3.2.2.1) to the K chosen harmonics at each EMT time step.

We will use the Fourier series theorem which states that a periodic signal s of period T can be approximated by: $s(t) \approx \sum_{k=-\infty}^{\infty} c_k(s) e^{2\frac{kj\pi t}{T}}$ with $c_k(s) = \frac{1}{T} \int_T s(\tau) e^{-2\frac{kj\pi\tau}{T}} d\tau$ is the k^{th} harmonic of the signal s .

We choose to keep only K harmonics, I is the set of harmonics chosen. The previous approximation is symmetric, so if the harmonic a is in I , the harmonic $-a$ (which is the complex conjugate of a) is also in I .

So the recombination of a periodic signal s of period T will be given by

$$s(t) \approx \sum_{k \in I} c_k(s) e^{2\frac{kj\pi t}{T}}.$$

However, when translating from EMT to dynamic phasor (3.2.1.1), a rolling history is used. With this moving history, an event has repercussions long after it ends. Indeed, as the figure 3.2 shows, an EMT event may still be in the history long after the event has ended, and may therefore still have an effect on the translation of EMT to TS.

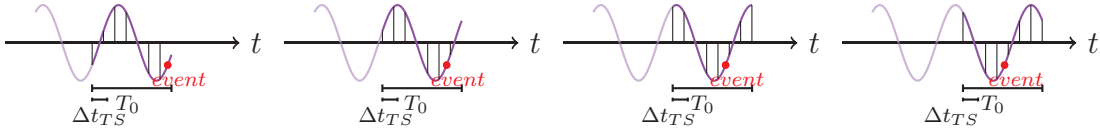


Figure 3.2: Impact of an EMT event in the History

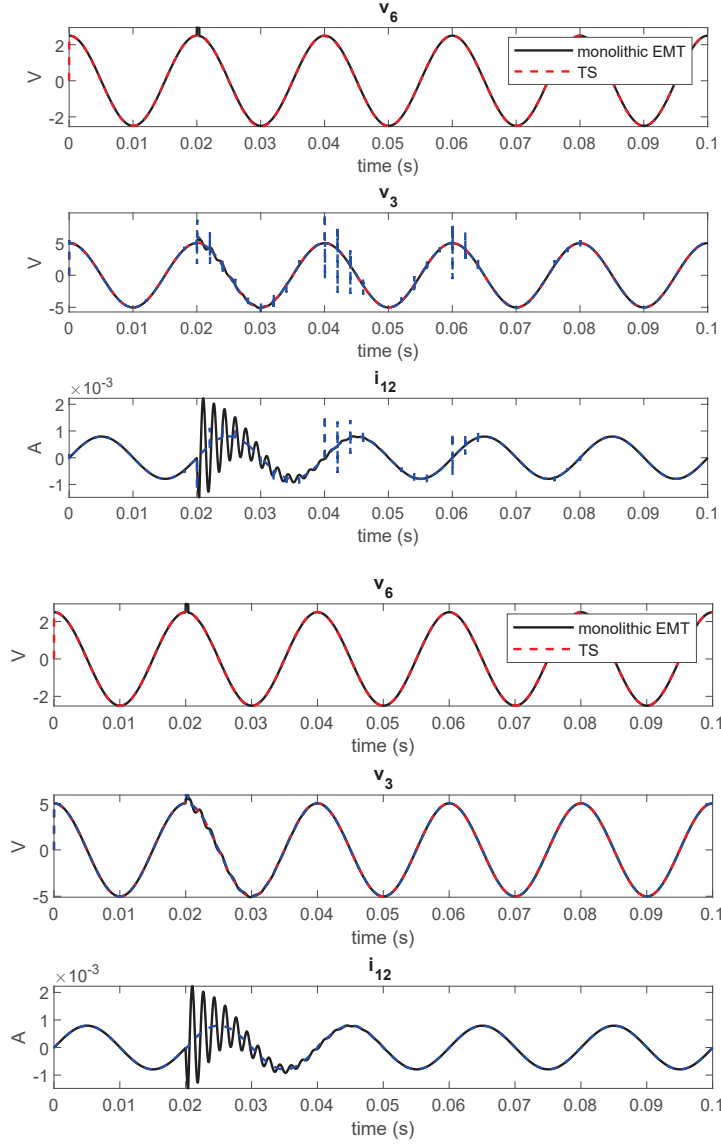


Figure 3.3: TS-EMT hybrid simulation of the test of section 2.5 with a jump in the voltage source at $t = 0.2s$ that last less than Δt_{TS} : without (top) and with (bottom) linear interpolation of \mathbb{W}_{TS}^{N+1}

Due to the jump between the first two rows of $\mathbb{W}_{i,e_{ts}}^{N+1}$ in the equation 3.6, and the repercussions of an event remained in the history of the EMT, there are leaps in the solution on the side of the EMT. Indeed, these repercussions have almost no impact on the dynamic phasor part, but they have a significant impact on the EMT part. The fact that it is visible on the EMT part is due to the Gibbs phenomenon. Indeed when there is a discontinuity jump, the Fourier sums (on which the translation

of TS into EMT is based) reveal the Gibbs phenomenon which is expressed in the form of an overshoot at this jump then of swings. Gottlieb and Shu [48] provide insight into this phenomenon as well as a way to obtain accurate function approximations, despite the presence of Gibbs' phenomenon, using filters. On this work the choice is made to smooth the first jump (the one between $(x^n, y^n)^t$ and $W_{i,e_{ts}}^{N+1}(t^{n+1})$) which is not a physical jump but a digital jump, in order not to have the appearance of this phenomenon.

As can be seen in the figure 3.3 there is, for the values calculated in EMT, an imitation of the phenomena of the event with an attenuation at each period, and this until the event is out of the history.

To smooth out these repercussions, the (3.2.2.1) is applied to a linear interpolation of two successive time steps of TS. α the number of EMT time steps over which the TS values are smoothed, $\alpha \in [1, m]$ with m the number of EMT time steps during a TS time step, and n as $T^N = t^n$.

Definition 9. \mathbb{T}_{emt}^{ts} , the translation operator from TS to EMT is defined as:

$$\mathbb{T}_{emt}^{ts}(t^l)W_{i,e_{ts}}^{N+1} = \begin{cases} \sum_{k \in I} \left(\frac{n+\alpha-l}{\alpha-1} w_{k;i,e_{ts}}^N + \frac{n+1-l}{1-\alpha} w_{k;i,e_{ts}}^{N+1} \right) e^{jk\omega_0 t^l} & \text{if } l \in [n+1, n+\alpha] \\ \sum_{k \in I} w_{k;i,e_{ts}}^{N+1} e^{jk\omega_0 t^l} & \text{if } l \in [n+\alpha, n+m] \end{cases}$$

3.2.3 Initialization

Correct initialization is important in our method, first of all, because if we start from a solution that is too far from the true solution, more iterations will be needed to obtain the true solution (if we do not use the acceleration method of the Aitken convergence). Bad initial conditions can also be considered as an event, an undesirable physical disturbance.

The difference with the usual methods is that in this heterogeneous method, the initial conditions do not impact only the terms due to the discretization. Indeed, this correct initialization is particularly important in our heterogeneous case because of the translation operators used.

- For the EMT to TS translation operator: in order to be able to start the translation from the EMT to TS, a history of the size of an already pre-filled period is necessary. It is therefore necessary to initialize over a period in time step EMT. With the moving history, the initial conditions can be present in the history on several macro time steps, thus these initial conditions have a diffuse impact on several time steps and not only on the initial moment of the simulation.
- For the operator of translation of TS towards EMT: A linear interpolation is used between two successive macro time steps, for the first time step the impact of the initial condition is thus very strong.

However, it is recalled that one of the motivations of this co-simulation is to gain in computational efficiency, which is why it would make no sense to have an initialization that is costly in terms of computation time.

To meet these requirements, two choices are available to us: the first is to calculate the monolithic steady-state simulation over a full period with a time step equal to the EMT time step. One recovers the real part of the solution in steady state at each step of time to fill the history. The second possibility is to calculate the monolithic steady-state simulation over a complete period with a time step equal to the time step TS and to apply a linear extrapolation for each time step EMT to fill the history.

For a TS domain $W_{i,ts}$ the EMT history initialization condition is given by:

$$Z_{0,ie_{emt}} = [\Re(z_{ie_{steadystate}}(0)), \dots, \Re(z_{ie_{steadystate}}(\tilde{m}\Delta t_{emt}))]^t.$$

To be coherent the TS domain $W_{i,ts}$ is initialize as $w_{0,i_{ts}} = \mathbb{T}_{ts}^{emt} Z_{0,ie_{emt}}$

For an EMT subdomain $W_{d,emt}$, only the first value of $\mathbb{W}_{d,emt}^1$ needs to be initialized, and this initialization is given by: $w_{0,d,emt} = \Re(z_{d,steadystate}(\tilde{m}\Delta t_{emt}))$. We also initialize $\mathbb{W}_{de_{ts}}$: $W_{0,de_{ts}}(t^l) = \sum_{k \in I} w_{0,k;d,e_{ts}} e^{jk\omega_0 t^l}$ for $l \in [0, m]$

3.2.4 Heterogeneous EMT-TS RAS formulation

We have a local writing of the Schwarz algorithm for the different EMT and TS subsystems. We also have the initialization of the problem. We can therefore write the RAS:

Definition 10. *The $p+1$ iteration of the RAS can be written for two subdomains as globally as follows:*

$$\begin{cases} w_{1_{ts}}^{N+1,p+1} &= \mathbb{A}_{1_{ts}}^{-1} (\mathbb{I}_{1;1e_{ts}} w_{1,1e}^{N,\infty} - \mathbb{E}_{1e_{ts}}^{emt} Z_{1e_{emt}}^{N+1,p} + G_{1_{ts}}^{N+1}), \\ \mathbb{W}_{2_{emt}}^{N+1,p+1} &= \mathbb{H}_{2_{emt}}^{-1} (\mathbb{E}_{2_{emt}}^{ts} \mathbb{W}_{2,e_{ts}}^{N+1,p} + G_{2_{emt}}^{N+1}) \end{cases} \quad (3.8)$$

Remark 8. *It can be observed that the system obtained is very similar to the one obtained in the homogeneous case, and that in order for the error operator to remain linear, the operators \mathbb{T}_{ts}^{emt} and \mathbb{T}_{emt}^{ts} must be linear.*

3.3 Heterogeneous EMT-TS RAS error operator and Acceleration of convergence

The system (3.8) is iterated at each time step until the error between two successive iterations reaches the tolerance. In this section we will discuss the impact of the heterogeneity of the co-simulation on the convergence of the method and on the acceleration strategy.

3.3.1 Acceleration strategy

There are several way to use the Aitken's acceleration technique. As seen in the chapter 2 we can or compute the error operator in the whole domain and accelerate the whole solution at once, or compute the error operator only on artificial boundaries accelerate only the interface values and use this converged values to obtain the whole solution with local resolutions. In the chapter 2 we computed the error algebraically but this operator can also be computed numerically. Indeed, Tromeur-Dervout [110, 111] developed a completely algebraic formulation of this Aitken's technique for accelerating convergence as a function of the trace of the Schwarz's iterations on the domain decomposition interface. To construct the error operator corresponding to n values, it is necessary to perform $n + 1$ Schwarz iterations. For this reason, most of the time, the error operator will be built only to speed up the values of the artificial boundaries. The way to compute the error operator numerically is as follow:

3.3.1.1 Numerical Computation of the error Operator

We define the global interface Γ as the concatenation of the $W_{i,e}^p$, i.e $\Gamma = \{W_{0,e}^p, \dots, W_{N-1,e}^p\}$ of size $n_\Gamma = \sum_{i=0}^{N-1} n_{i,e}$.

Proposition 7. *The operator $P \in \mathbb{R}^{n_\Gamma \times n_\Gamma}$ can be computed algebraically after $n_\Gamma + 1$ iteration, if the matrix $[e^{n_\Gamma}, \dots, e^1]$ is non-singular as:*

$$P = [e^{n_\Gamma+1}, \dots, e^2][e^{n_\Gamma}, \dots, e^1]^{-1}. \quad (3.9)$$

with e^{k+1} the error between two consecutive RAS iterations $e^{k+1} = z_\Gamma^{k+1} - z_\Gamma^k$.

Proof. The error between two RAS iterations. As P does not depend on the RAS iteration:

$$\begin{aligned} e^1 &= Pe^0 \\ e^2 &= Pe^1 \\ e^3 &= Pe^2 \\ &\vdots \\ e^{p+1} &= Pe^p \end{aligned}$$

By concatenating the successive equations we have the result. \square

This operator can be used to speed up convergence. This error operator should only be calculated for the first time step. Unless the topology changes or there is a non-linearity in which case the operator must be recomputed.

3.3.1.2 Impact of heterogeneity in the choice of acceleration strategy

In the homogeneous case of the chapter 2 the error operator was computed numerically, but it is more complicated to compute the error operator analytically in the heterogeneous case, due to the translation operators. We will then numerically calculate this error operator.

We seek to accelerate convergence at the interfaces between two subdomains, a TS $W_{1_{ts}}$ and an EMT $W_{2_{emt}}$. We will therefore first accelerate the convergence of the interfaces: $Z_{1_{e_{emt}}}$ and $\mathbb{W}_{2_{e_{ts}}}$.

Let's calculate the size of the interface values, note $n_{i,e}$ the size of interface i, e :

First let's calculate for $Z_{1_{e_{emt}}}$ we just need to speed up the values taken at the last macro time step, the "Active Boundary Conditions": $Z_{1_{e_{emt}}}$ so the sample size to be accelerated is $n_{1,e} \times m$ (with $\Delta t_{ts} = m\Delta t_{emt}$).

Then let's calculate the size of the interface $\mathbb{W}_{2_{e_{ts}}}$, $2K \times n_{2,e}$, (with K the number of harmonics that we chose to keep).

So the whole interface between these two sub-domains is of size $n_{1,e} \times m + 2K \times n_{2,e}$, so to speed up this interface it will be necessary to realize $n_{1,e} \times m + 2K \times n_{2,e} + 1$ Schwarz iteration, which can be huge. Indeed, m is often around a hundred, K is rarely greater than 2. We cannot afford to do so many iterations of Schwarz. Two solutions are available to us:

- Accelerate the values coming from the EMT after translation, in this way, the size of the whole interface to be accelerated will be $2K \times n_{1,e} + 2K \times n_{2,e}$.
- One can choose to accelerate only the interface values coming from the TS side, i.e to compute the error operator P linked only to the $\mathbb{W}_{2_{e_{ts}}}$ values. Like so, an error operator of size $2K \times n_{2,e}$ will be computed by performing $2K \times n_{2,e} + 1$ iterations. Once the true values of $\mathbb{W}_{2_{e_{ts}}}$ are obtained, they are used to compute the local resolution of the EMT part $\mathbb{W}_{2_{emt}}$. These EMT values are in turn used for the local resolution of the TS part $w_{1_{ts}}^{N+1,p+1}$.

We'll prefer the second option because it further limits the number of Schwarz iterations needed.

3.3.2 Error: a multifactorial resultant

As seen in the homogeneous case the RAS error depends on the value of components (above all the inductance and capacitance values), but also depends on the topology and the size of the time steps. We can wonder the impact of the heterogeneity on the convergence. There are some more factor possible: the ratio

between EMT and TS time steps, the localization of some components (in an EMT or in a TS subdomain), and finally the impact of the translation operator in the convergence.

3.3.2.1 Impact of the translation operator from EMT to TS

As we choose to keep some harmonics and therefore remove some information, we can expect the translation to decrease the error. Let's calculate the error on the TS side to see the impact of rolling history on the error:

$$\begin{aligned} w_{1ts}^{N+1,p+1} - w_{1ts}^{N+1,\infty} &= \mathbb{A}_{1ts}^{-1}(-\mathbb{E}_{1e_{ts}}^{emt}(Z_{1e_{emt}}^{N+1,p} - Z_{1e_{emt}}^{N+1,\infty-1})), \\ w_{1ts}^{N+1,p+1} - w_{1ts}^{N+1,\infty} &= \mathbb{A}_{1ts}^{-1}(-\mathbb{E}_{1e_{ts}}^{emt}[0, \dots, 0, \underbrace{z_{ie_{emt}}^{m \times N+1,p}, \dots, z_{ie_{emt}}^{m \times (N+1),p}]}_{\mathbb{W}_{1e_{emt}}^p - \mathbb{W}_{1e_{emt}}^{\infty-1}}]t), \end{aligned}$$

Since only active boundary conditions will have an impact, therefore having a moving history also has an impact on the error operator.

Remark 9. *In the numerical results, it will be difficult to quantify the impact of mobile history on the error. Indeed, the impact of the mobile history will be inseparable from the impact of the reduction of the macro time steps Δt_{ts} which already greatly reduces the error.*

3.3.2.2 Impact of the translation operator from TS to EMT

When recombining harmonics, there is no change in the level of information, so the recombination itself will not change the error. On the other hand, smoothing can have an impact.

Let's calculate the EMT rated error to see the impact of smoothing on the error:

$$\mathbb{W}_{2emt}^{N+1,p+1} - \mathbb{W}_{2emt}^{N+1,\infty} = \mathbb{H}_{2emt}^{-1} \mathbb{E}_{2emt}^{ts} (\mathbb{W}_{2,e_{ts}}^{N+1,p} - \mathbb{W}_{2,e_{ts}}^{N+1,\infty-1}).$$

For $l \in [n+1, n+\alpha]$ we can write:

$$\begin{aligned} W_{2,e_{ts}}^{N+1,p}(t^l) - W_{2,e_{ts}}^{N+1,\infty-1}(t^l) &= \sum_{k \in I} \left(\frac{n+\alpha-l}{\alpha-1} w_{k;2,e_{ts}}^N + \frac{n+1-l}{1-\alpha} w_{k;2,e_{ts}}^{N+1,p} \right) e^{jk\omega_0 t^l} \\ &\quad - \sum_{k \in I} \left(\frac{n+\alpha-l}{\alpha-1} w_{k;2,e_{ts}}^N + \frac{n+1-l}{1-\alpha} w_{k;2,e_{ts}}^{N+1,\infty} \right) e^{jk\omega_0 t^l} \\ W_{2,e_{ts}}^{N+1,p}(t^l) - W_{2,e_{ts}}^{N+1,\infty-1}(t^l) &= \sum_{k \in I} \frac{n+1-l}{1-\alpha} (w_{k;2,e_{ts}}^{N+1,p} - w_{k;2,e_{ts}}^{N+1,\infty}) e^{jk\omega_0 t^l}. \end{aligned}$$

We will therefore have

$$\mathbb{W}_{2_{emt}}^{N+1,p+1} - \mathbb{W}_{2_{emt}}^{N+1,\infty} = \mathbb{H}_{2_{emt}}^{-1} \mathbb{E}_{2_{emt}}^{ts} \begin{pmatrix} 0 \\ \frac{0}{1-\alpha} \\ \vdots \\ \frac{1-\alpha}{1-\alpha} \\ 1 \\ \vdots \\ 1 \end{pmatrix} (\mathbb{W}_{2_{e_{ts}}}^{N+1,p} - \mathbb{W}_{2_{e_{ts}}}^{N+1,\infty-1}).$$

So there is a smoothing of $\frac{n+1-l}{1-\alpha} \in [0, 1]$ on the first α values of each macro step, we expect a behavior of reduction of the radius of convergence, ie the method will converge less quickly or will diverge less quickly. It is expected that the translation operator from TS to EMT has no impact on the linearity of the error, indeed this operator being a combination of two linear operators is linear.

3.4 Heterogeneous EMT-TS Numerical Results

For these numerical results, we use the same framework as in the homogeneous case (cf chapter 2). The same circuit is therefore used, with the same cutouts (see Figure 2.2). First, we study numerically the convergence of the method. As the RAS method diverged in the homogeneous case we can expect the same kind of results. Secondly, we study the results obtained by the heterogeneous co-simulation in order to see the gain of the EMT-TS modeling over the TS modeling.

3.4.1 Heterogeneous EMT-TS RAS convergence results

3.4.1.1 Effect of circuit topology on the convergence

In order to see only the effect of the circuit topology (i.e the effect of the circuit components values) on the convergence, we use the classic FFT to translate from EMT to TS keeping Δt_{ts} to be a period. We also do not smooth the TS to EMT translation either, but simply recombine the harmonics (ie $\alpha = 0$). The impact on the convergence of the method of these modifications to the translation operators will be given in a second time.

We know that the value of the components influences the convergence of the method ([91]), and more particularly the inductance and the capacitance have a very strong impact.

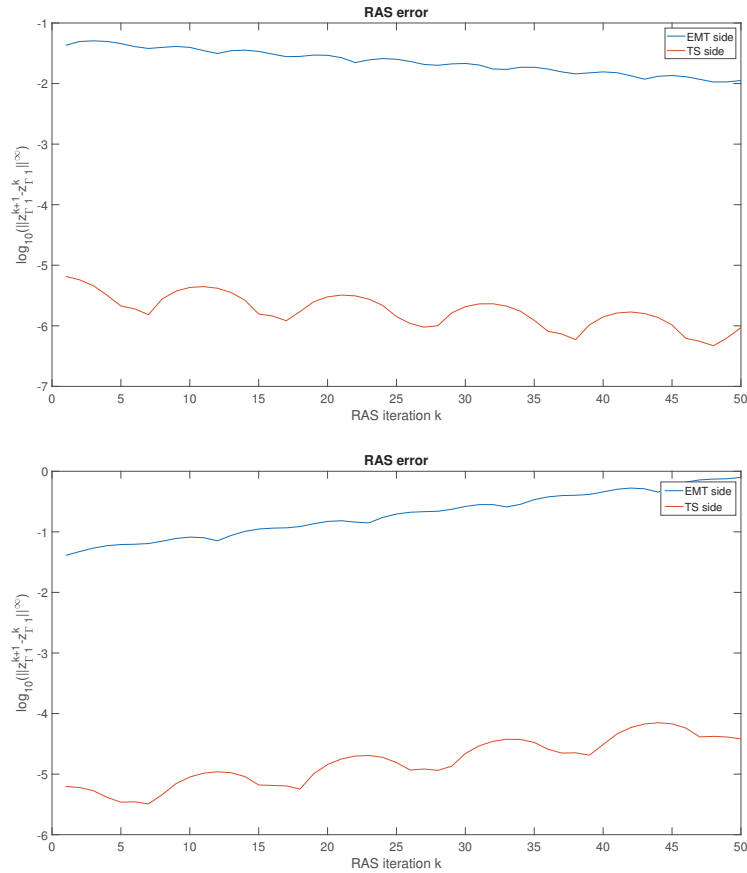


Figure 3.4: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with $C_1 = C_2 = 1.10^{-6}$, $R_1 = R_2 = 7, L_1 = 0.07$ and we have $L_2 = 1$ at the top and $L_2 = 0.07$ at the bottom.

Figure 3.4 shows the error between two heterogeneous EMT-TS RAS iterations by changing the value of the inductance L_2 in the circuit. It exhibits a convergent method for $L_2 = 1$ while the method diverges for $L_2 = 0.07$ the other components being set. Figure 3.5 shows the error between two heterogeneous EMT-TS RAS iterations by changing the values of the capacitances C_1 and C_2 in the circuit. It exhibits a convergent method for $C_1 = C_2 = 1.10^{-6}$ while the method strongly diverges for $C_1 = C_2 = 1.10^{-4}$ the other components being set.

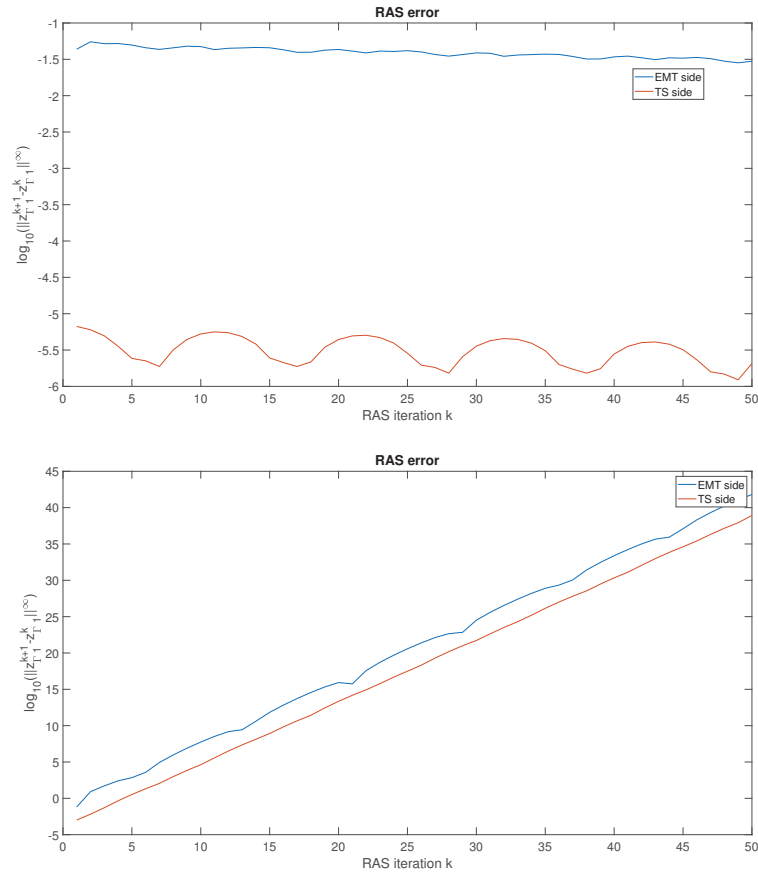


Figure 3.5: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.4$ and we have $C_1 = C_2 = 1.10^{-6}$ at the top and $C_1 = C_2 = 1.10^{-4}$ at the bottom.

The choice of the part that is simulated with the TS or with the EMT also has an impact and more particularly the place where the voltage source is located has an impact in the convergence of the method.

Figure 3.6, gives the error between two consecutive heterogeneous EMT-TS RAS iterations according to whether the voltage source is modeled within the EMT (left) or within the TS (right). It exhibits a divergent method when the source is in the EMT part while the method is convergent if it is located in the TS part.

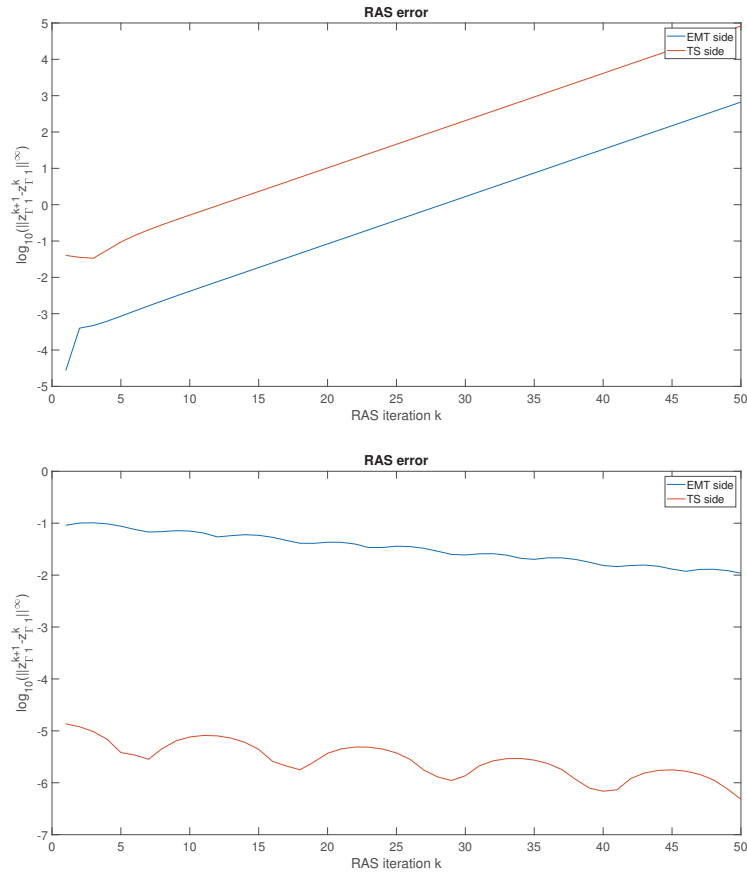


Figure 3.6: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$. At the top the source is in the EMT subsystem and at the bottom the source is in the TS subsystem.

These results clearly demonstrate the need for the heterogeneous EMT-TS RAS method to have the Aitken's acceleration technique like in the homogeneous RAS case to be independent of the circuit topology.

3.4.1.2 Effect of the EMT Time steps Δt_{emt} on the convergence

We have seen with Figure 3.6 that the convergence depends on the subsystem in which the source is located. However, the impact of the localization of the voltage source is coupled to the size of the EMT time steps. Indeed, a modification of the size of the EMT time steps does not have the same impact depending on the location of the voltage source.

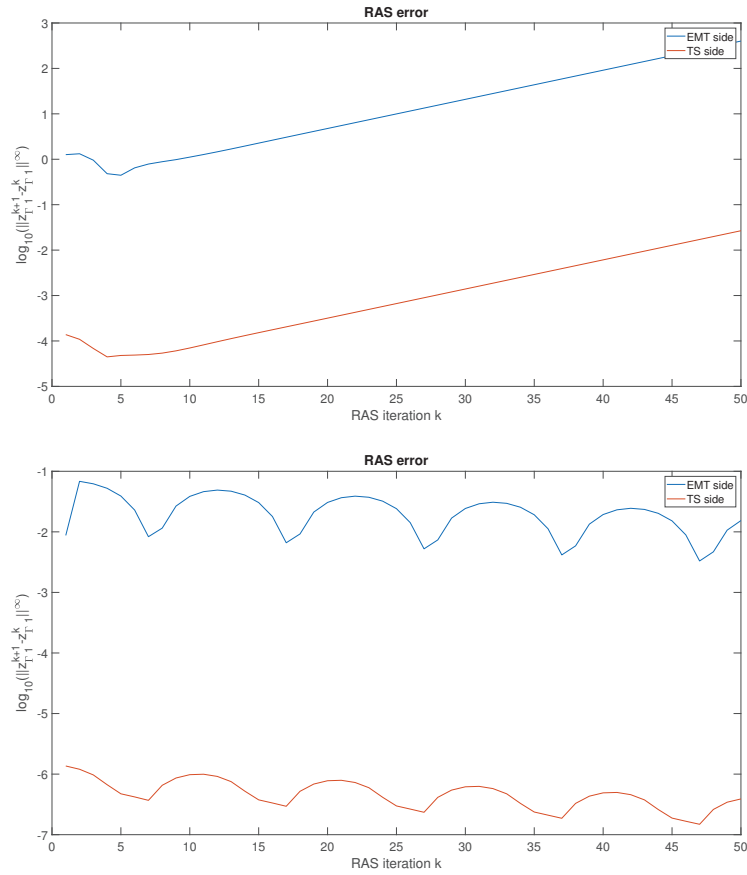


Figure 3.7: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$ at the top $\Delta t_{empt} = 2.10^{-3}$ and at the bottom $\Delta t_{empt} = 2.10^{-5}$.

Figure 3.7 gives the error between two consecutive heterogeneous RAS iterations with respect of the value of Δt_{empt} for the circuit problem while the source is modeled in TS. It exhibits that the method diverges with $\Delta t_{empt} = 2.10^{-3}$ (left) while the method converges for $\Delta t_{empt} = 2.10^{-5}$ (right).

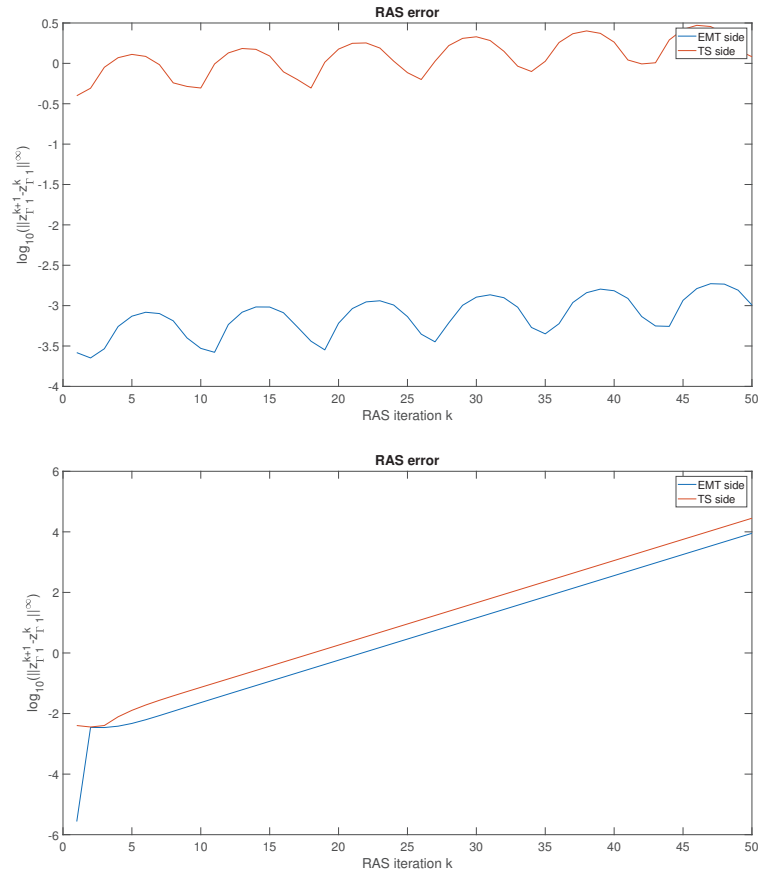


Figure 3.8: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the EMT subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$ at the top $\Delta t_{emt} = 2.10^{-3}$ and at the bottom $\Delta t_{emt} = 2.10^{-5}$.

Figure 3.8 gives the error between two consecutive heterogeneous RAS iterations with respect of the value of Δt_{emt} for the circuit problem when the source is modeled in EMT. It exhibits that the method diverges for both cases $\Delta t_{emt} = 2.10^{-3}$ (top) and $\Delta t_{emt} = 2.10^{-5}$ (bottom). Indeed the more Δt_{emt} decreases the more the error increases.

These results clearly demonstrate the need for the heterogeneous EMT-TS RAS method to have the Aitken's acceleration technique as in the homogeneous RAS case to be independent of the time step size choice.

3.4.1.3 Effect of the translation operators on the convergence

Let us study the influence on the convergence of the percentage α in the smoothing during the translation from TS to EMT. We recall that it represents the number of EMT time steps on which the interpolation of TS interface values are performed.

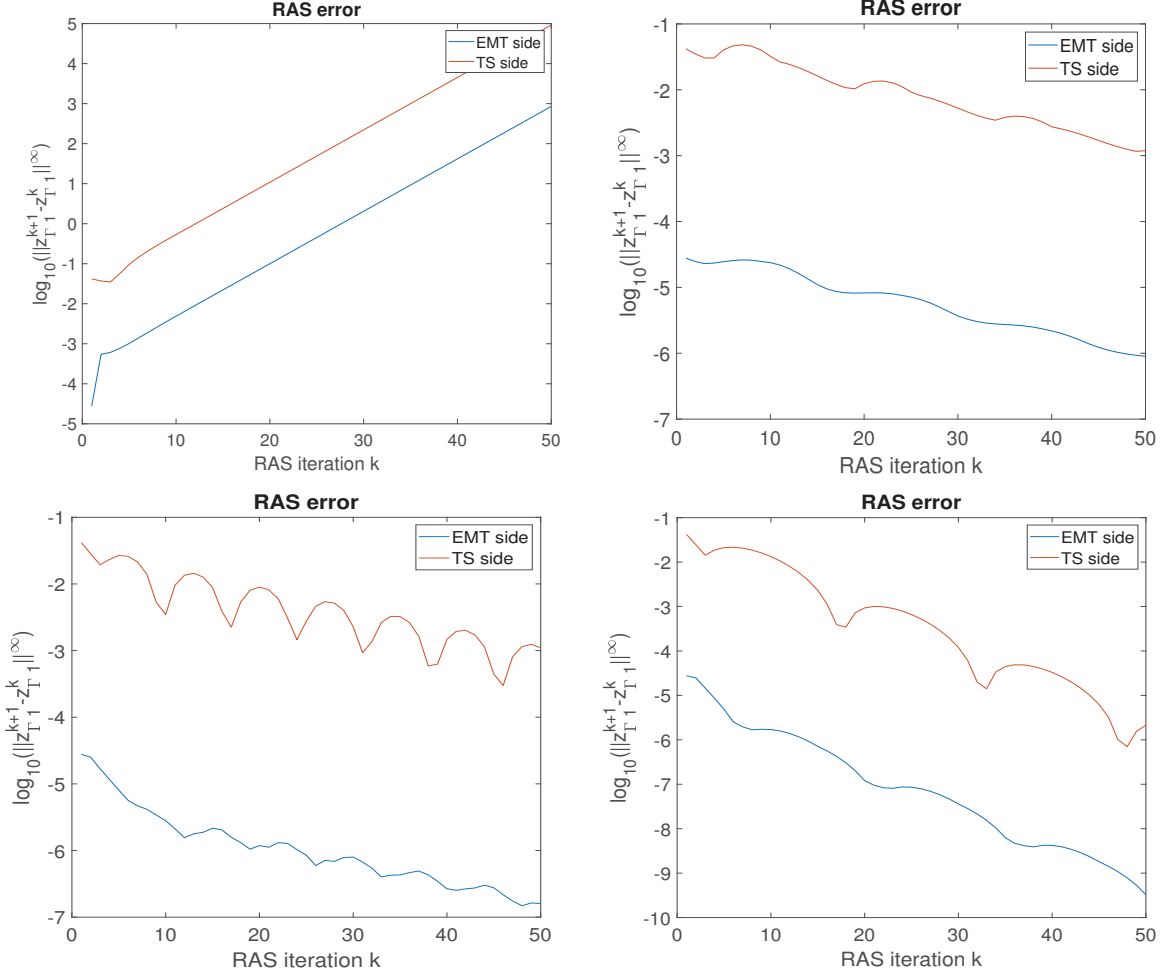


Figure 3.9: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the α parameter in the smoothing and with voltage source in the EMT subsystem. $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$. With top left $\alpha = 0\%$, top right $\alpha = 25\%$, bottom left $\alpha = 50\%$ and bottom right $\alpha = 75\%$

Figure 3.9 shows the error between two consecutive heterogeneous EMT-TS RAS iterations for $\alpha = \{0\%, 25\%, 50\%, 75\%\}$. It exhibits that the α has a strong effect on the convergence as the method diverges when $\alpha = 0\%$ (no smoothing) while

it converges in other cases. It also shows that the larger is α , the better is the convergence.

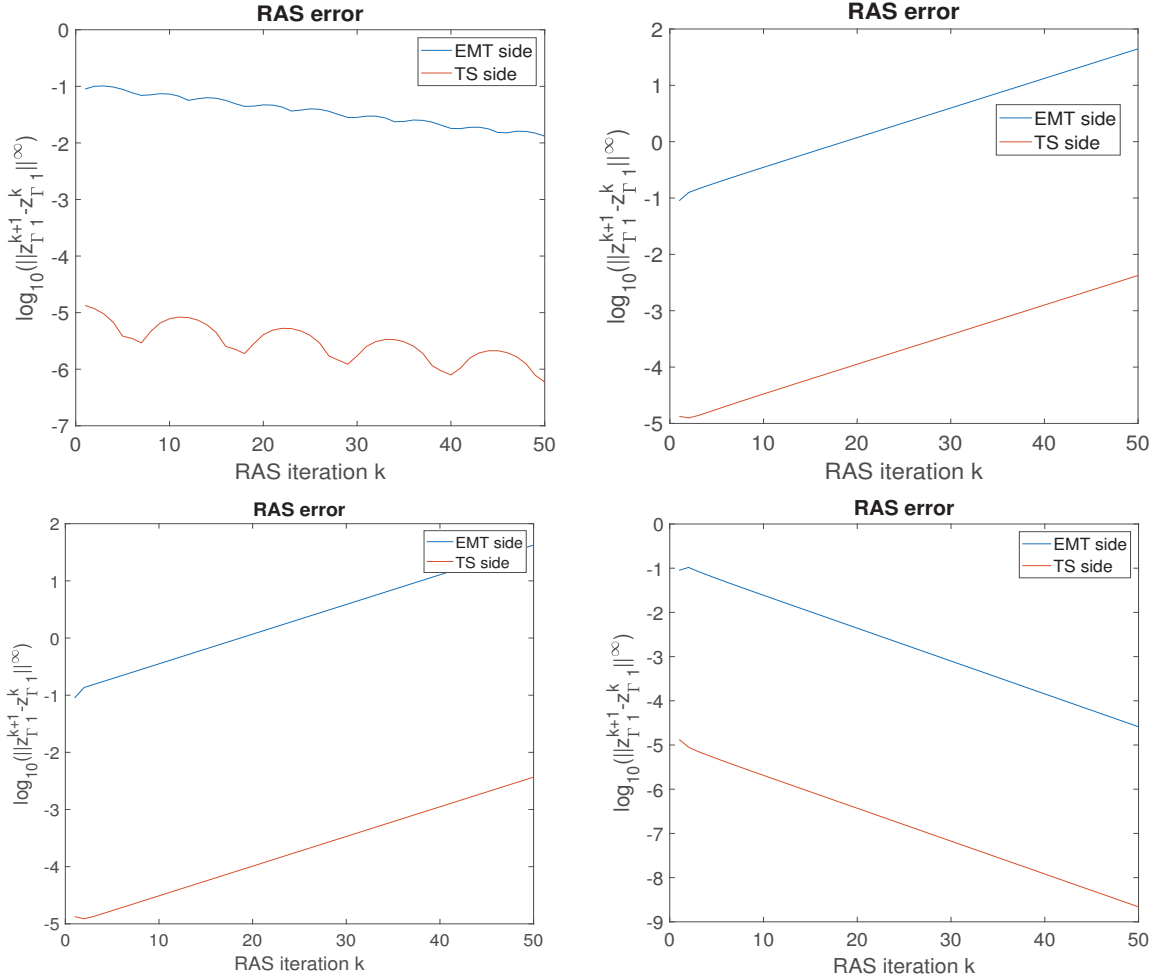


Figure 3.10: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the α parameter in the smoothing and with voltage source in the TS subsystem. With, $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$. With top left $\alpha = 0$, top right $\alpha = 25$, bottom left $\alpha = 50$ and bottom right $\alpha = 75$

As the increase in α has a beneficial effect on convergence in the case where the voltage source is in the EMT part, we investigated whether this will also be the case for the case where the voltage source is in the TS part which was convergent without smoothing. Figure 3.10 shows the error between two heterogeneous EMT-TS RAS iterations by varying α when the source is in the TS part. It exhibits

that the convergence deteriorates when α becomes larger than 0% until α reaches a threshold (about 50%). Once this threshold is exceeded, the error decreases with the increase of α . We notice that the convergence improves faster than it has deteriorated.

In conclusion, smoothing has a beneficial impact on the convergence of the heterogeneous EMT-TS RAS method as we found α values that make the method convergent regardless of the source voltage location. Nevertheless, the convergence behaviour is not a monotonic function with respect to α .

3.4.1.4 Effect of the reduction of Δ_{ts} and moving history on the convergence

Finally, let us study the effect of the reduction of the time steps Δ_{ts} with respect to the period T and the moving history on the convergence.

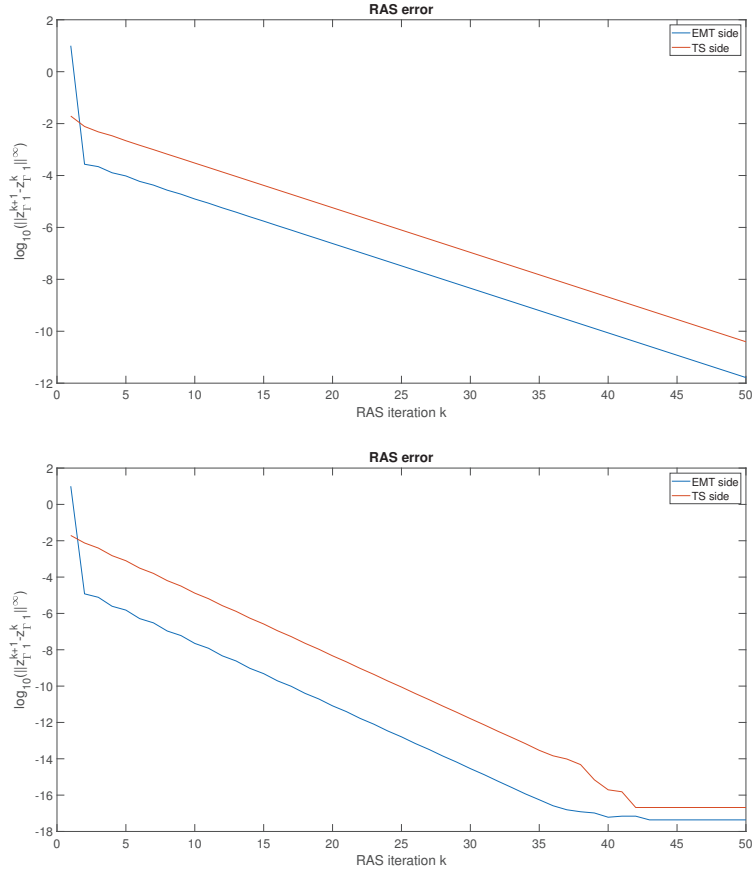


Figure 3.11: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the $\Delta_{ts} = 10^{-2}$ less than a the period T and rolling history, with the source voltage in EMT and $\alpha = 0\%$ (top), $\alpha = 25\%$ (bottom), $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$.

Figure 3.11 shows the error between two consecutive heterogeneous EMT-TS RAS iterations for $\Delta_{ts} = 1/(2T)$, $\alpha = \{0\%, 25\%\}$ and the voltage source in EMT. It exhibits that the decrease of $\Delta_{ts} = 1/(2T)$ and moving history lead to a convergent method in comparison of Figure 3.9 (top left) with $\Delta_{ts} = 1/(T)$ which was divergent. The decrease of Δ_{ts} as also a beneficial impact on the convergence associated to $\alpha = 25\%$ with an error nearby 10^{-12} for 50 iterations for $\Delta_{ts} = 10^{-2}$ instead of 10^{-6} for $\Delta_{ts} = 2 \cdot 10^{-2}$ (Figure 3.11 (right) in comparison of Figure 3.9 (top right)).

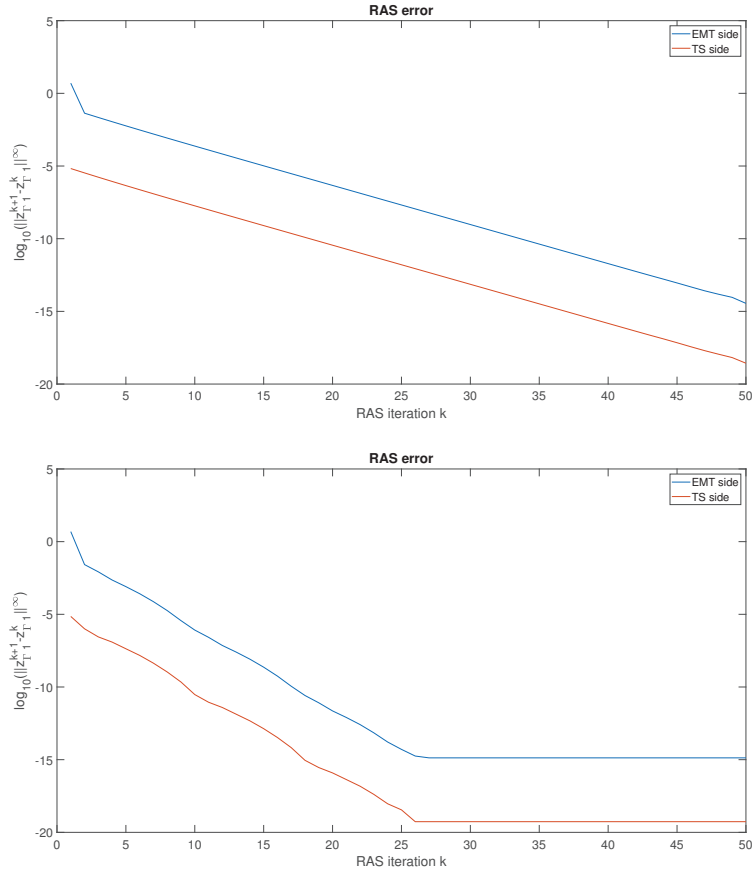


Figure 3.12: Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the $\Delta_{ts} = \Delta_{ts} = 10^{-2}$ less than a the period T and rolling history, with the source voltage in TS and $\alpha = 0\%$ (top), $\alpha = 25\%$ (bottom), $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$.

Figure 3.12 shows the error between two consecutive heterogeneous EMT-TS RAS iterations for $\Delta_{ts} = \frac{1}{2T}$, $\alpha = \{0\%, 25\%\}$ and the voltage source in TS. It also exhibits the beneficial effect of the decrease of $\Delta_{ts} = \frac{1}{2T}$ and moving history as the

method converge for $\alpha = 25\%$ that was not the case for $\Delta t_{ts} = \frac{1}{T}$ in Figure 3.10 (top right). The convergence is also greatly improved for the $\alpha = 0\%$ case with an error of 10^{-15} in 40 iterations versus an error of 10^{-6} for $\Delta t_{ts} = \frac{1}{T}$ in Figure 3.10 (top left).

In conclusion, decreasing the Δt_{ts} time step associated with the moving history has greatly improved the convergence of the heterogeneous EMT-TS RAS, and made it less dependent on the source voltage location. Nevertheless, the decrease in Δt_{ts} also decreases the ratio of Δt_{ts} to Δt_{emt} and thus the potential performance of the heterogeneous EMT-TS RAS.

3.4.1.5 Largest eigenvalue of the heterogeneous EMT-TS RAS error operator

In order to confirm the convergence behaviour of the heterogeneous EMT-TS RAS, we numerically calculated the error operator P of the method and its largest eigenvalue.

Source in the EMT subsystem				
$\alpha \backslash \Delta t_{ts}$	2.10^{-2}	$1.5 \cdot 10^{-2}$	$1. \cdot 10^{-2}$	$2. \cdot 10^{-3}$
0%	$\lambda(P) = -1.3882$	$\lambda(P) = -0.735 \pm 0.015i$	$\lambda(P) = -0.7068$	$\lambda(P) = -0.2537$
25%	$\lambda(P) = -0.884 \pm 0.407i$	$\lambda(P) = -0.7231$	$\lambda(P) = -0.6050$	$\lambda(P) = -0.2538$
50%	$\lambda(P) = -0.876 \pm 0.427i$	$\lambda(P) = -0.6447$	$\lambda(P) = -0.4592$	$\lambda(P) = -0.2523$
75%	$\lambda(P) = -0.812 \pm 0.197i$	$\lambda(P) = -0.4938$	$\lambda(P) = 0.3633$	$\lambda(P) = -0.2503$
Source in the TS subsystem				
$\alpha \backslash \Delta t_{ts}$	$2. \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$	$1. \cdot 10^{-2}$	$2. \cdot 10^{-3}$
0%	$\lambda(P) = -1.010 \pm 0.290i$	$\lambda(P) = -1.0953$	$\lambda(P) = -0.5383$	$\lambda(P) = -0.1716$
25%	$\lambda(P) = -1.1430$	$\lambda(P) = -1.0417$	$\lambda(P) = -0.4478$	$\lambda(P) = -0.1413$
50%	$\lambda(P) = -1.1789$	$\lambda(P) = -0.8259$	$\lambda(P) = -0.282 \pm 0.053i$	$\lambda(P) = -0.1033$
75%	$\lambda(P) = -0.8930$	$\lambda(P) = -0.503 \pm 0.043i$	$\lambda(P) = -0.209 \pm 0.083i$	$\lambda(P) = -0.0831$

Table 3.1: Largest eigenvalue of the error operator $P_{\Gamma_{ts}}$, with, $R_1 = R_2 = 7, L_1 = L_2 = 0.07, C_1 = C_2 = 1.10^{-6}$ and $\Delta t_{emt} = 2.10^{-4}$. At top with the source in the EMT subsystem and at bottom in the TS subsystem

Table 3.1 gives the combined impact of smoothing and moving history on the convergence of the method by showing the value of the largest eigenvalue of the error operator P for source voltage in EMT and TS. It exhibits that except when we are in the particular case where the source of tension is in TS and that $\Delta t_{ts} = 1/T$, the increase of α always improves the convergence. The decrease of Δt_{ts} ,

possible thanks to the moving history always improves the convergence too. The combination of the two makes it possible to obtain fairly rapid convergences but still too much from an operational point of view. This is why we need to accelerate its convergence.

3.4.1.6 Aitken's acceleration of the heterogeneous EMT-TS RAS convergence

For all the studies of convergences of the previous sub-section, we see that the convergence/divergence is purely linear, we will therefore use the method of acceleration of convergence of Aitken to obtain the solution after $n_T + 1$ iterations.

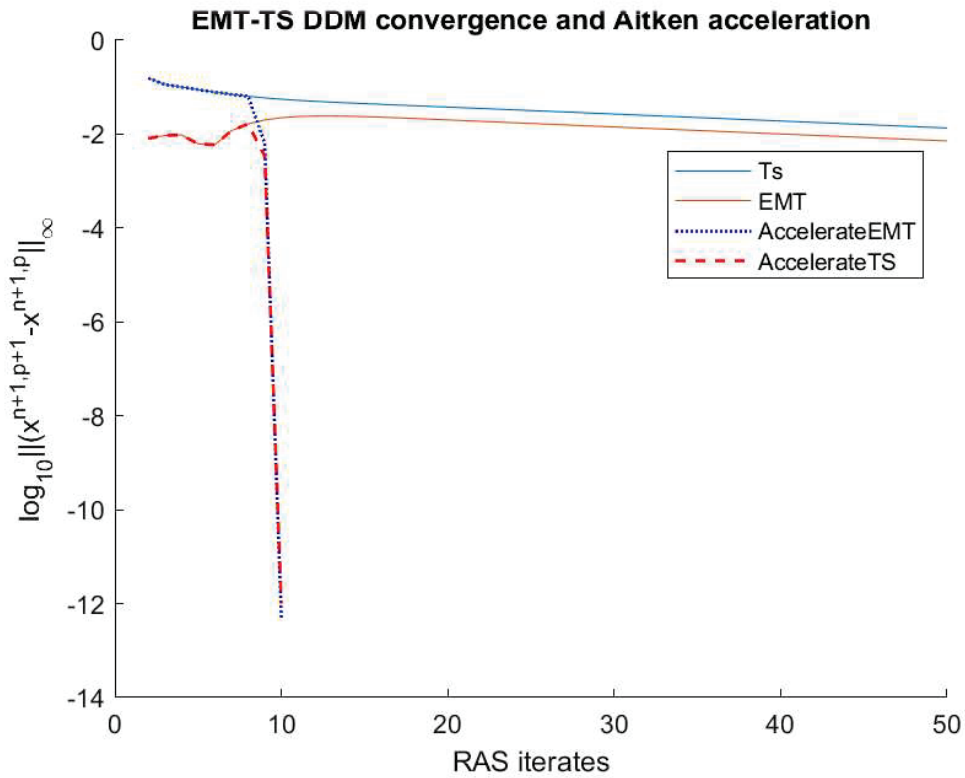


Figure 3.13: Heterogeneous EMT-TS RAS convergence error for for each subdomain, for the time step $t = 0.02$ and its Aitken's acceleration applied on the TS partition interface with $\Delta t_{ts} = 2.10^{-2}$ and $\Delta t_{emt} = 2.10^{-4}$) and with parameters $L_1 = 0.07$, $C1 = 1.10^{-6}$, $R1 = 7$, $L2 = 0.07$, $C2 = 1.10^{-6}$, $R2 = 7$, $Zs = 0.000001$, with the voltage source in the TS part

Figure 3.13 gives the \log_{10} of the error between two consecutive RAS iterations at

time $t = 0.02$. It shows a linear convergence behavior and can therefore be accelerated by the Aitken acceleration of the convergence technique after 9 iterations needed to numerically construct the error operator P_T . In this case the method is convergent and it is shown that after the local resolutions, the accelerated method has converged on each subdomain.

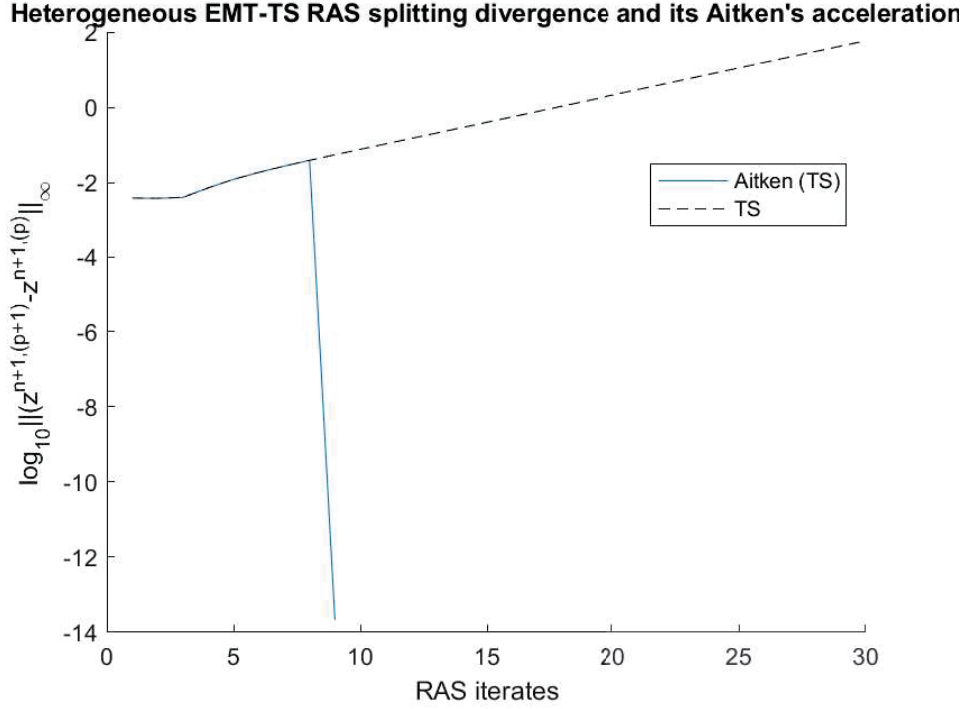


Figure 3.14: Heterogeneous EMT-TS RAS convergence error for the TS boundary for the time step $t = 0.02$ and its Aitken's acceleration applied to the TS partition interface with $\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$) and with parameters $L_1 = 0.07$, $C1 = 1.10^{-6}$, $R1 = 7$, $L2 = 0.07$, $C2 = 1.10^{-6}$, $R2 = 7$, $Z_s = 0.000001$, with the voltage source in the EMT part

Figure 3.14 shows the \log_{10} of the error between two successive heterogeneous EMT-TS RAS iterations with and without the use of the Aitken acceleration method with the voltage source in the EMT part with $\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$). It exhibits that the method diverges but as the divergence is purely linear the Aitken's acceleration of the convergence technique succeeds to retrieve the true solution after 9 iterations as the acceleration is performed on the interface solution of the TS part. Figure 3.15 shows the solution after using the Aitken acceleration technique in a divergent case. The result is compared to the monolithic EMT case showing that the true solution is obtained.

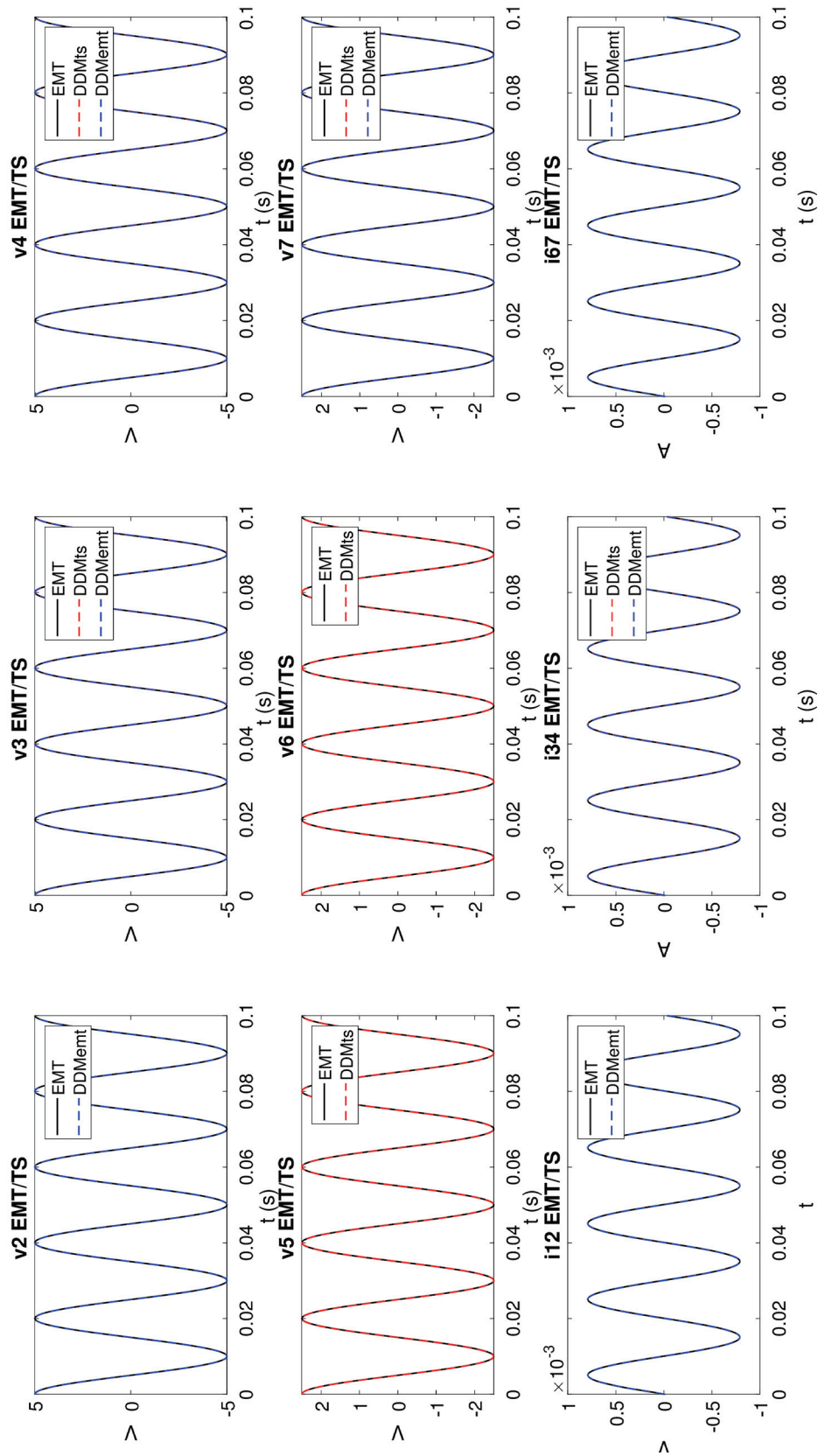


Figure 3.15: Heterogeneous EMT-TS RAS solution comparison with the monolithic EMT case on a time intervalle with $\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$) and with parameters $L_1 = 0.07$, $C_1 = 1.10^{-6}$, $R_1 = 7$, $L_2 = 0.07$, $C_2 = 1.10^{-6}$, $R_2 = 7$, $Z_s = 0.000001$, with the voltage source in the EMT part

3.4.2 Qualitative results on the heterogeneous EMT-TS RAS

The numerical tests that follow focus on the qualitative advantage of the EMT-TS model over the TS model. The problem is that of the circuit of figure 2.2.

3.4.2.1 The advantage of the EMT part

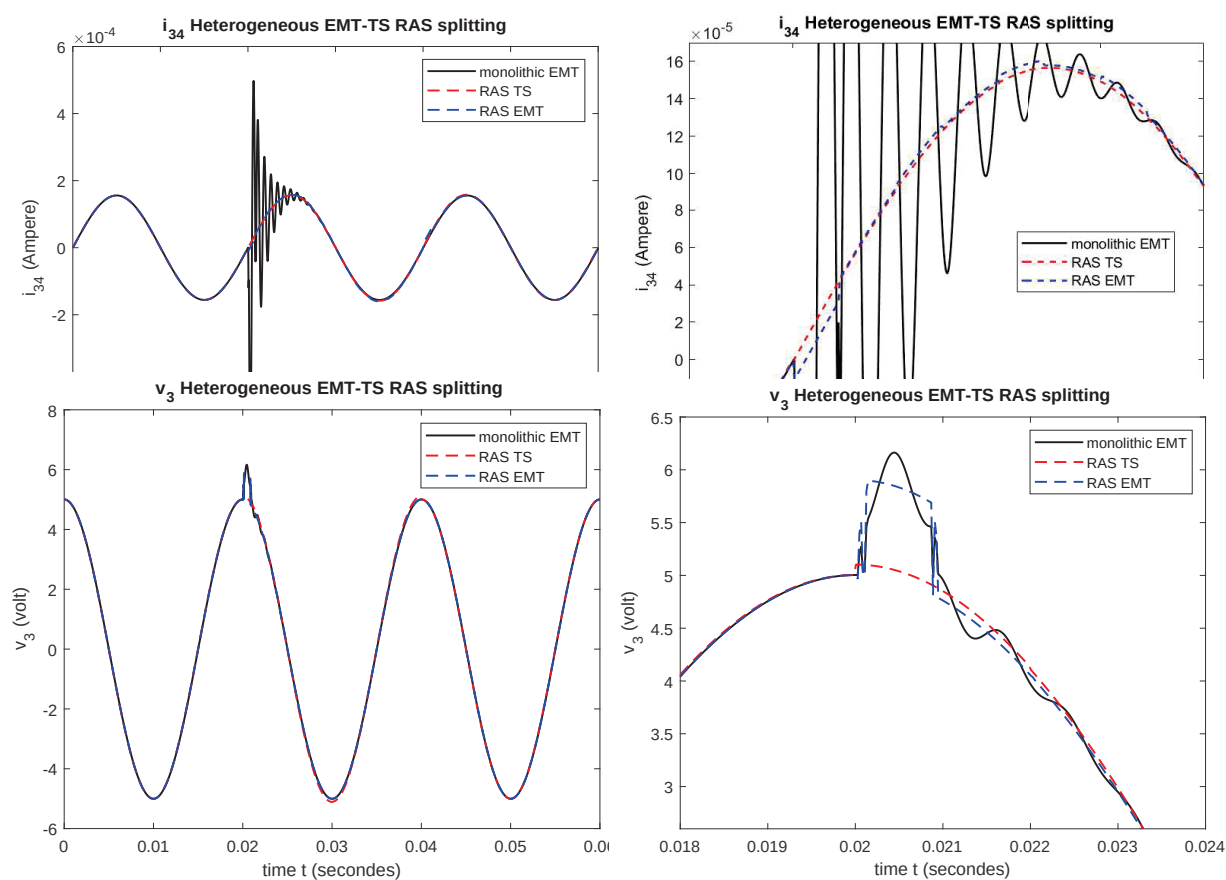


Figure 3.16: Comparison of the behavior, with respect to time, of the variables i_{34} (top left) and v_3 (bottom left) (the figures on the right are their zoom on the disturbances) computed using the heterogeneous EMT-TS RAS splitting with the Aitken's technique for accelerating convergence ($\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$), the reference is the monolithic EMT. An amplitude perturbation on the voltage source starting at $t = 0.02$ s and ending at $t = 0.021$ s, therefore lasting less than one Δt_{ts} is applied. Parameters are $L_1 = 0.07$, $C_1 = 1.10^{-5}$, $R_1 = 7$, $L_2 = 0.07$, $C_2 = 1.10^{-7}$, $R_2 = 7$, $Z_s = 0.000001$.

The purpose of having a part of the circuit simulated in the EMT modelling is to capture events that are not visible with the dynamic phasor modeling. In order to verify that the EMT-TS modeling can capture this advantage, we will create a disturbance that lasts less than a TS time step and therefore cannot be seen by the TS modeling.

Figure 3.16 compares the EMT monolithic reference values for the variables v_3 and i_{34} with the EMT-TS heterogeneous RAS splitting where a perturbation on the source voltage that starts at $t = 0.02\text{s}$ and ends at $t = 0.021\text{s}$ is applied. It exhibits that the heterogeneous EMT-TS RAS succeeds in capturing part of the perturbation on the v_3 . It shows a good agreement between the monolithic and the heterogeneous EMT-TS RAS for the variable v_3 . The variable i_{34} in the EMT DDM part captures certain oscillations due to the perturbation. These results show that heterogeneous EMT-TS RAS can capture disturbances that last less than one TS time step and therefore would not have been captured by a monolithic TS model.

Remark 10. *These results are visible because we chose a dynamic phasor rather than a simple phasor modeling for the TS part.*

3.4.2.2 The impact of the cutting and passing of information

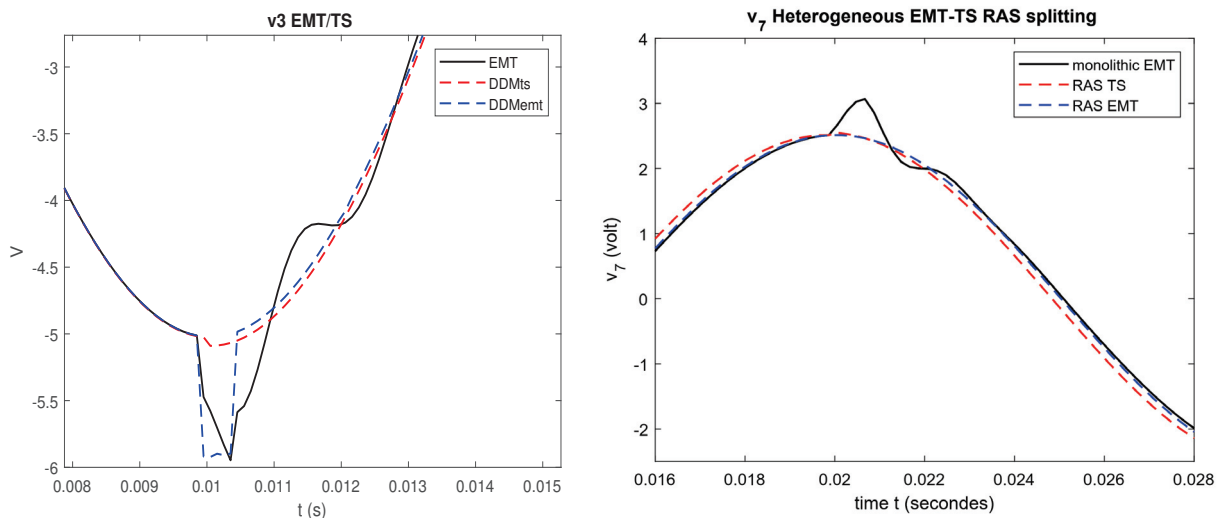


Figure 3.17: Comparison of the behavior, with respect to time, of the variables v_3 (left) and v_7 (right) computed using the heterogeneous EMT-TS RAS splitting with the Aitken's technique for accelerating convergence ($\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{empt} = 2.10^{-5}$), the reference is the monolithic EMT. An amplitude perturbation on the voltage source. Parameters are $L_1 = 0.07$, $C_1 = 1.10^{-5}$, $R_1 = 7$, $L_2 = 0.07$, $C_2 = 1.10^{-7}$, $R_2 = 7$, $Z_s = 0.000001$.

For the figure 3.17 a short event at the voltage source is created. Two potentials equidistant from the voltage source are compared v_3 and v_7 , it is noted that the solutions found by the EMT subdomain during the cosimulation capture the disturbance much better and are closer to the monolithic solution for the potential v_3 compared to the potential v_7 . Note that v_3 is close to the transition from EMT to TS, while v_7 is close to a boundary where information is transmitted only from TS to EMT. This gives us an indication of the loss of information implied by the TS part.

3.5 Conclusions

In this chapter, we have set up a co-simulation between two types of models: Transient Electromagnetics and Dynamic Phasors. To do this, we used a Schwarz method. We have also implemented translation operators between the two types of models. The linearity of these translation operators makes it possible to keep the pure convergence of the method. This pure linear convergence/divergence of the method allows us to accelerate the convergence to the true solution using Aitken's convergence acceleration method. We have numerically constructed the error operator associated with the interface from the RAS iterations. We used an RLC circuit to obtain numerical results, we could observe that the part calculated in EMT has results very close to the monolytic EMT reference. We have also been able to see that disturbances which would have been invisible by a monolytic TS simulation are slightly perceived by the TS part of the heterogeneous simulation. There is therefore a gain between level of precision and calculation time. These results open perspectives on the representation of two models on the overlap, we have in the numerical results an overlap but a larger model would allow a better comparison of the results given by the two different models on the overlap. Indeed, it will allow to give more details on the loss of information during the transition from EMT to TS and thus to work on the boundary conditions and on the partitioning.

Chapter 4

Dynamic Iteration

Contents

4.1	Introduction	80
4.1.1	Motivations	80
4.1.2	Previous works for accelerating Dynamic iteration	80
4.2	Dynamic Iteration	81
4.3	Dynamic Iteration with RAS splitting	86
4.3.1	DI with RAS splitting in the continuous case	86
4.3.2	Discrete counterpart of the DI with RAS splitting	89
4.4	Time stepping strategies for DI	90
4.4.1	Sequential DI strategy	90
4.4.2	Pipelined DI strategy	91
4.4.3	Link between the error operators of sequential DI and pipelined DI	93
4.4.4	Comparison between Sequential and pipelined strategy	95
4.5	Numerical Results on DI	98
4.5.1	Sequential strategy	98
4.5.2	Pipelined DI with RAS splitting strategy	100
4.6	Conclusion	103

4.1 Introduction

4.1.1 Motivations

The purpose of this chapter is to exhibit the link between the domain decomposition method we developed in the previous chapters with the class of methods named waveform relaxation or dynamic iteration methods. We are going to show that our domain decomposition belongs to this class by considering a special splitting based on the restricted additive Schwarz. We can then apply the theory developed for dynamic iteration and adapt some theoretical results with our specific splitting. As the dynamic iteration is mainly viewed as a fixed point method its convergence depends on the spectral radius of the error operator depending itself on the splitting used. We will thus be able to show that purely linear convergence or divergence results can be reused in the context of the linear DAE system, thus allowing us to accelerate convergence towards the true solution with the Aitken technique to accelerate convergence. Using the context of dynamic iteration, we develop two different strategies (step by step acceleration and multisteps acceleration) for applying the Aitken's acceleration of the convergence technique. We show the link between the two strategies and we will discuss the cases where each strategy is the most optimal. Numerical results confirm the advantages of the dynamic iteration accelerated with Aitken's acceleration even in some nonlinear and non-uniform time steps cases.

4.1.2 Previous works for accelerating Dynamic iteration

Since the pioneering work of Lelarasme & al [65] that analyze in time domain large-scale problems arising from the modeling of integrated circuits, waveform relaxation methods (WR) [68] also known as dynamic iteration methods, a term first introduced by Miekkala and Nevanlinna [78, Eq (2.2)] and generally used in publications [77, 11, 17, 19, 49] arouses more and more interest with the development of parallel computers [101] and more generally in the co-simulation framework [19, 104].

In such methods applied to Ordinary Differential Equations (ODE) systems or to Differential Algebraic Equations (DAE) systems, the system is decomposed into several subsystems with many internal variables and few external variables. For initial value problems with linear ODEs, the method consists in carrying out some splitting of the linear operator $A = M - N$ [78] such as Jacobi, relaxed Jacobi, Gauss-Seidel or SOR. Nevertheless this fixed-point process must be contractant to converge. The analysis of the convergence of the method, using the Laplace transform, occurs when the spectral radius $\max_{\xi \in \mathbb{R}} \rho((i\xi I + M)^{-1}N) < 1$ [78, Eq 2.13]. For initial value problems with linear DAE systems $B\dot{x} + Ax = f$, like

those arising in RLC circuits, Miekkala [77, Theorem 2] extended her previous convergence analysis result with the splitting of $B = M_b - N_b$ and $A = M_A - N_A$, $\max_{\xi \in \mathbb{R}} \rho((i\xi M_B + M_A)^{-1}(i\xi N_B + N_A)) < 1$. Reichel & al combined the waveform SOR with the multistep integration method and showed that the SOR relaxation optimal parameter is dependent on Fourier frequencies [101, Eq 19]. Jiang and Wing determined the expressions of the spectrum and pseudospectrum of the waveform relaxation operators for linear differential-algebraic equations systems which occur especially in circuit simulation [58, Eqs (3) & (4)] and Jiang extended these results to a general class of nonlinear differential-algebraic equations [57] of index one, these extended results generalize the expressions of Lumsdaine and Wu[69]. Several techniques to precondition the fixed point problem were proposed by Arnold & Gunther [12]. Hout has established convergence results that are relevant in applications to nonlinear, nonautonomous, stiff initial value problems [54].

Some convergence acceleration techniques for the WR have been proposed. Some waveform successive overrelaxation (SOR) techniques have been proposed by Janssen and Vandewalle [56] to accelerate the standard waveform method. Leimkuhler proposed to accelerate the WR by solving the defect equations with a larger timestep, or by using a recursive procedure based on a succession of increasing timesteps [64]. Lumsdaine & Wu proposed to accelerate the WR by Krylov subspace techniques (WGMRES) [70] to solve time-dependent problems. Botchev & al [22] compared WR-Krylov with Krylov's methods combined with the shift and invert (SAI) technique to obtain parallelism in time. Ladics [62] combined the WR with convergent numerical methods to solve semi-linear PDEs, he showed the effect of applying time windows. Recent developments in the dynamic iteration method for the co-simulation of electrical circuits have been carried out by Bartel & al [18, 17] and by Ali & al [2]. Gausling & al [46] analyzed the contraction and the rate of convergence of the co-simulation process for a test circuit subjected to uncertainties on the parameters of its components. Moreover, the rate of convergence or divergence of the dynamic iteration depends on the interface coupling [45]. Pade and Tischendorf [91] presented topological criteria for the coupling of networks which are easy to check and which are sufficient to ensure the convergence of the WR which is related to the DAE index.

4.2 Dynamic Iteration

Let us recall the method of dynamic iteration applied to the DAE as it was described by Miekkala[77]:

Consider the initial linear problem (4.1) to find a function $x : [0, T] \rightarrow \mathbb{C}^n$

$$\begin{cases} B\dot{x}(t) + Ax(t) = f(t), t \in]0, T] \\ x(0) = x_0. \end{cases} \quad (4.1)$$

with B and A being $\mathbb{C}^{n \times n}$ matrices, B can be singular (so this system would be a system of differential algebraic equations), and $f : [0 : T] \rightarrow \mathbb{C}^n$ is a known input function.

The dynamic iteration is an iterative scheme to solve Eq. (4.1) based on the splitting of operators $A = M_A - N_A$ and $B = M_B - N_B$, where M_B is assumed to be invertible.

Definition 11 (Dynamic iteration). *The dynamic iteration scheme at iteration k to solve Eq. (4.1) with the splitting $A = M_A - N_A$ and $B = M_B - N_B$ and starting from an initial guess $x^{(0)}$ satisfying $x^{(0)}(0) = x_0$ is written as follows:*

$$\begin{cases} M_B\dot{x}^{(k)}(t) + M_Ax^{(k)}(t) = N_B\dot{x}^{(k-1)}(t) + N_Ax^{(k-1)}(t) + f(t), t \in]0, T] \\ x^{(k)}(0) = x_0. \end{cases} \quad (4.2)$$

Proposition 8 (Miekkala). *If M_B is chosen invertible then there exist a linear operator P independant of iteration k such as*

$$x^{(k)}(t) = P_t x^{(k-1)}(t) + \phi(t) \quad (4.3)$$

where

$$Pu(t) = M_B^{-1}N_Bu(t) + \int_0^t e^{-M_B^{-1}M_A(s-t)}M_B^{-1}(N_A - M_AM_B^{-1}N_B)u(s)ds \quad (4.4)$$

$$\phi(t) = e^{-M_B^{-1}M_At}(I - M_B^{-1}N_B)x_0 + \int_0^t e^{-M_B^{-1}M_A(s-t)}M_B^{-1}f(s)ds \quad (4.5)$$

Let us retrieve the proof of Eq. (4.3) not detailed in [77].

Proof. As M_B is considered invertible, we can write Eq. from (4.2)

$$\begin{aligned} \dot{x}^{(k)} - M_B^{-1}N_B\dot{x}^{(k-1)} &= -M_B^{-1}M_A(x^{(k)} - M_B^{-1}N_Bx^{(k-1)}) \\ &\quad + M_B^{-1}(N_A - M_AM_B^{-1}N_B)x^{(k-1)} + M_B^{-1}f \end{aligned} \quad (4.6)$$

By proceeding to a change of variable $z = x^{(k)} - M_B^{-1}N_Bx^{(k-1)}$, the solution of the homogeneous equation associated to Eq. (4.6),

$$z_h(t) = e^{-M_B^{-1}M_At} \quad (4.7)$$

The general solution is written $z(t) = z_h(t)\psi(t)$. By putting $z(t)$ in Eq. (4.6)

$$\begin{aligned} -M_B^{-1}M_Ae^{-M_B^{-1}M_At}\psi(t) + e^{-M_B^{-1}M_At}\dot{\psi}(t) &= -M_B^{-1}M_Ae^{-M_B^{-1}M_At}\psi(t) \\ &\quad + M_B^{-1}(N_A - M_AM_B^{-1}N_B)x^{(k-1)}(t) + M_B^{-1}f(t) \end{aligned}$$

we find that:

$$\dot{\psi}(t) = e^{M_B^{-1}M_A t}(M_B^{-1}(N_A - M_A M_B^{-1}N_B)x^{(k-1)}(t) + M_B^{-1}f(t)) \quad (4.8)$$

Then by integrating $\dot{\psi}(t)$

$$\psi(t) = \int_0^t e^{M_B^{-1}M_A s}(M_B^{-1}(N_A - M_A M_B^{-1}N_B)x^{(k-1)}(s) + M_B^{-1}f(s))ds + Cst$$

The Cst is defined such that $x^{(k)}(0) = x^{(k-1)}(0) = x_0$ i.e. $z(0) = (I - M_B^{-1}N_B)x_0$.

We obtain the solution $z(t)$ to be:

$$\begin{aligned} z(t) &= e^{-M_B^{-1}M_A t} \int_0^t e^{M_B^{-1}M_A s}(M_B^{-1}(N_A - M_A M_B^{-1}N_B)x^{(k-1)}(s))ds \\ &\quad + e^{-M_B^{-1}M_A t} \left(\int_0^t e^{M_B^{-1}M_A s} M_B^{-1}f(s) ds \right) + (I - M_B^{-1}N_B)x_0 \\ x^{(k)}(t) - M_B^{-1}N_B x^{(k-1)}(t) &= \int_0^t e^{M_B^{-1}M_A(s-t)}(M_B^{-1}(N_A - M_A M_B^{-1}N_B)x^{(k-1)}(s))ds \\ &\quad + \int_0^t e^{M_B^{-1}M_A(s-t)} M_B^{-1}f(s) ds + e^{-M_B^{-1}M_A t}(I - M_B^{-1}N_B)x_0 \end{aligned}$$

finally, we obtain the result with the linear operator P acting on $x^{(k-1)}$ and the function $\phi(t)$ independant of $x^{(k-1)}$:

$$\begin{aligned} x^{(k)} &= \underbrace{M_B^{-1}N_B x^{(k-1)}(t) + \int_0^t e^{M_B^{-1}M_A(s-t)}(M_B^{-1}(N_A - M_A M_B^{-1}N_B)x^{(k-1)}(s))ds}_{P_t x^{(k-1)}} \\ &\quad + \underbrace{e^{-M_B^{-1}M_A t}(I - M_B^{-1}N_B)x_0 + \int_0^t e^{M_B^{-1}M_A(s-t)} M_B^{-1}f(s) ds}_{\phi(t)} \end{aligned}$$

□

Remark 11. *The linear operator P_t depends on the time t . It is also the error operator of the DI as $\phi(t)$ depends uniquely on x_0 and $f(t)$ and does not depend on the iteration k . The dynamic iteration defined by 4.2 converges if the spectral radius of P_t is such as : $\rho(P_t) < 1$. This is the drawback of the method as the operator P_t depends on the time t we need that the $\rho(P_t) < 1$ for all $t \in]0, T]$.*

The extension of the dynamic iteration to DAE system have been conducted. Jiang & Wing [58], obtained the same type of results for linear DAE systems written in the state space form. Let $M \in \mathbb{C}^{n_1 \times n_1}$, $A \in \mathbb{C}^{n_1 \times n_1}$, $B \in \mathbb{C}^{n_1 \times n_a}$, $C \in \mathbb{C}^{n_a \times n_1}$,

$D \in \mathbb{C}^{n_a \times n_a}$ matrices and $f_1 :]0, T] \rightarrow \mathbb{C}^{n_1}$, $f_2 :]0, T] \rightarrow \mathbb{C}^{n_a}$ functions, $x_0 \in \mathbb{C}^{n_1}$ initial state value defining the DAE system in state-space form Eq. (2.17):

$$\begin{cases} M\dot{x}(t) + Ax(t) + By(t) &= f_1(t), t \in [0, T], \\ Cx(t) + Ny(t) &= f_2(t), t \in [0, T], \\ x(0) &= x_0, \end{cases} \quad (4.9)$$

Where $x :]0, T] \rightarrow \mathbb{C}^{n_1}$ are the n_1 searched state solutions and $y :]0, T] \rightarrow \mathbb{C}^{n_a}$ are the n_a searched algebraical solutions.

Definition 12 (Dynamic Iteration for linear DAE). *The Dynamic Iteration scheme for Eq. (4.9) considers the splitting of matrices M, A, B, C, N as $M = M_1 - M_2, A = A_1 - A_2, B = B_1 - B_2, C = C_1 - C_2, N = N_1 - N_2$, where matrices M_1 and N_1 are assumed non-singular (which implies that the DAE system has index one)*

$$\begin{cases} M_1\dot{x}^{(k)}(t) + A_1x^{(k)}(t) + B_1y^{(k)}(t) &= M_2\dot{x}^{(k-1)}(t) + A_2x^{(k-1)}(t) + B_2y^{(k-1)}(t) \\ &+ f_1(t), t \in [0, T] \\ C_1x^{(k)}(t) + N_1y^{(k)}(t) &= C_2x^{(k-1)}(t) + N_2y^{(k-1)}(t) \\ &+ f_2(t), t \in [0, T], \\ x^{(k)}(0) &= x_0, \end{cases} \quad (4.10)$$

Afterwards, Jiang & Wing defined an iteration operator for Dynamic Iteration applied to DAE system similar to the iteration operator P_t of Miekkala for ODE system:

Theorem 1 (Jiang & Wing [58]). *By grouping the differential and algebraic variables in the same vector $z = [x, y]^t$, the system 4.10 can be written with $\phi(t) = [\phi_1(t), \phi_2(t)]^t$ as:*

$$\begin{aligned} z^{(k)}(t) &= P_t z^{(k-1)}(t) + \phi(t) \\ P_t &= \begin{pmatrix} M_1^{-1}M_2 & 0 \\ N_1^{-1}S & N_1^{-1}N_2 \end{pmatrix} + \begin{pmatrix} \mathcal{R}_1^c & \mathcal{R}_2^c \\ -N_1^{-1}C_1\mathcal{R}_1^c & -N_1^{-1}C_1\mathcal{R}_2^c \end{pmatrix}. \end{aligned} \quad (4.11)$$

With

$$(\mathcal{R}_1^c u)(t) = \int_0^t e^{-M_1 D_1(t-s)} M_1^{-1} (D_2 - D_1 M_1^{-1} M_2) u(s) ds, \forall u \in L^2([0, T], \mathbb{C}^n) \quad (4.12)$$

$$(\mathcal{R}_2^c v)(t) = \int_0^t e^{-M_1^{-1} D_1(t-s)} M_1^{-1} (B_2 - B_1 N_1^{-1} N_2) v(s) ds, \forall v \in L^2([0, T], \mathbb{C}^n) \quad (4.13)$$

$$D_1 = A_1 - B_1 N_1^{-1} C_1, \quad (4.14)$$

$$D_2 = A_2 - B_1 N_1^{-1} C_2, \quad (4.15)$$

$$S = (C_2 - C_1 M_1^{-1} M_2), \quad (4.16)$$

$$\begin{aligned} \phi_1(t) &= e^{-M_1^{-1} D_1 t} (I - M_1^{-1} M_2) x_0 \\ &\quad + \int_0^t e^{-M_1^{-1} D_1(t-s)} M_1^{-1} (f_1(s) - B_1 N_1^{-1} f_2(s)) ds, \end{aligned} \quad (4.17)$$

$$\phi_2(t) = -N_1^{-1} C_1 \phi_1(t) + N_1^{-1} f_2(t). \quad (4.18)$$

Proof. Let us retrieve the proof of Theorem 1 not detailed in [58].

The idea is to eliminate the variable $y^{(k)}$ from the algebraical equation and to substitute its expression in the differential equation:

$$y^{(k)} = N_1^{-1} (C_2 x^{(k-1)} - C_1 x^{(k)} + N_2 y^{(k-1)} + f_2) \quad (4.19)$$

$$\begin{aligned} M_1 \dot{x}^{(k)} + \underbrace{(A_1 - B_1 N_1^{-1} C_1)}_{D_1} x^{(k)} &= M_2 \dot{x}^{(k)} + \underbrace{(A_2 - B_1 N_1^{-1} C_2)}_{D_2} x^{(k-1)} \\ &\quad + (B_2 - B_1 N_1^{-1} N_2) y^{(k-1)} + f_1 - B_1 N_1^{-1} f_2 \end{aligned}$$

Then we apply the same technique as for the Dynamic Iteration for ODE system with $M_B = M_1$, $N_B = M_2$, $M_A = D_1$ and $N_A = D_2$ and $f(t) = (B_2 - B_1 N_1^{-1} N_2) y^{(k-1)}(t) + f_1(t) - B_1 N_1^{-1} f_2(t)$ to obtain:

$$\begin{aligned} x^{(k)} &= M_1^{-1} M_2 x^{(k-1)} + \underbrace{\int_0^t e^{-M_1^{-1} D_1(s-t)} M_1^{-1} (D_2 - D_1 M_1^{-1} M_2) x^{(k-1)}(s) ds}_{(\mathcal{R}_1^c x^{(k-1)})(t)} \\ &\quad + \underbrace{\int_0^t e^{-M_1^{-1} D_1(s-t)} M_1^{-1} (B_2 - B_1 N_1^{-1} N_2) y^{(k-1)}(s) ds}_{(\mathcal{R}_2^c y^{(k-1)})(t)} \\ &\quad + \underbrace{e^{-M_1^{-1} D_1 t} (I - M_1^{-1} M_2) x_0 + \int_0^t e^{-M_1^{-1} D_1(s-t)} M_1^{-1} (f_1(s) - B_1 N_1^{-1} f_2(s)) ds}_{\phi_1(t)} \end{aligned}$$

Then we substitute $x^{(k)} = (M_1^{-1}M_2 + (\mathcal{R}_1^c))x^{(k-1)} + \mathcal{R}_2^c y^{(k-1)} + \phi_1$ in Eq. (4.19):

$$\begin{aligned} y^{(k)}(t) &= N_1^{-1}(C_2 - C_1 M_1^{-1} M_2)x^{(k-1)}(t) - (N_1^{-1}C_1 \mathcal{R}_1^c x^{(k-1)})(t) \\ &\quad + N_1^{-1}N_2 y^{(k-1)}(t) - (N_1^{-1}C_1 \mathcal{R}_2^c y^{(k-1)})(t) \\ &\quad - \underbrace{N_1^{-1}C_1 \phi_1(t) + N_1^{-1}f_2(t)}_{\phi_2(t)} \end{aligned}$$

□

This result is interesting because it shows the linear convergence of the dynamic iteration applied to DAE systems and therefore the possibility of accelerating its convergence using Aitken's method to accelerate convergence.

Remark 12. *Let us notice that the error operator P_t depends again on the time $t \in]0, T]$ and it acts on the $n_1 + n_1$ components. We are going to show in the next section that the RAS method developed in the previous chapter 2 can be considered as a dynamic iteration with a specific splitting of the operators A, B, C, N*

4.3 Dynamic Iteration with RAS splitting

Let us consider the system of DAE (4.9) coming from the modeling of an electrical network where we choose $M = I$. This choice corresponds to a change of variables on the voltage terms in order to get rid of the \mathbb{I} matrix present in the DAE systems of the chapters 2 and 3.

4.3.1 DI with RAS splitting in the continuous case

The k^{th} iteration of the restrictive additive Schwarz is written locally for the W_i^p partition as:

$$\begin{cases} \dot{z}_i^{(k)}(t) + \mathbb{A}_i z_i^{(k)}(t) = G_i(t) - \mathbb{E}_i z_{i,e}^{(k-1)}(t), & t \in [0, T], \\ x_i^{(k)}(0) = R_i^{p,d} x_0. \end{cases} \quad (4.20)$$

Our goal is to put this RAS iteration in the form of the Dynamic Iteration with a specific splitting of operators A, B, C, D . For this purpose, we separate the differential and algebraic variables belonging to the partition W_i^p . Then the k^{th} RAS iteration is written:

$$\begin{cases} \dot{x}_i^{(k)}(t) + A_i x_i^{(k)}(t) + B_i y_i^{(k)}(t) = b_i^b(t) - E_{i,d}^d x_{ie}^{(k-1)}(t) - E_{i,d}^a y_{ie}^{(k-1)}(t), \\ C_i x_i^{(k)}(t) + D_i y_i^{(k)}(t) = b_i^a(t) - E_{i,a}^d x_{ie}^{(k-1)}(t) - E_{i,a}^a y_{ie}^{(k-1)}(t), \\ x_i^{(k)}(0) = R_i^{p,d} x_0, \quad t \in [0, T]. \end{cases} \quad (4.21)$$

Where

$$\begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix} = \begin{pmatrix} R_i^{p,d} & 0_{n_{i_1} \times n_2} \\ 0_{n_{i_2} \times n_1} & R_i^{p,a} \end{pmatrix} \mathbb{A} \begin{pmatrix} (R_i^{p,d})^T & 0_{n_1 \times n_{i_2}} \\ 0_{n_2 \times n_{i_1}} & (R_i^{p,a})^T \end{pmatrix}, \quad (4.22)$$

$$\begin{pmatrix} E_{i,d}^d & E_{i,d}^a \\ E_{i,a}^d & E_{i,a}^a \end{pmatrix} = \begin{pmatrix} R_i^{p,d} & 0_{n_{i_1} \times n_2} \\ 0_{n_{i_2} \times n_1} & R_i^{p,a} \end{pmatrix} \mathbb{A} \begin{pmatrix} (R_{i,e}^{p,d})^T & 0_{n_1 \times n_{i_2}} \\ 0_{n_2 \times n_{i_1}} & (R_{i,e}^{p,a})^T \end{pmatrix}. \quad (4.23)$$

Operator $R_i^{p,d}$ (respectively $R_i^{p,a}$) is the restriction to the differential variables (respectively algebraical variables) of R_i^p . We define also $\tilde{R}_i^{0,d}$ and $\tilde{R}_i^{0,a}$ such that

$$\tilde{R}_i^0 = \begin{pmatrix} \tilde{R}_i^{0,d} & 0_{n_{i_1} \times n_2} \\ 0_{n_{i_2} \times n_1} & \tilde{R}_i^{0,a} \end{pmatrix}.$$

Proposition 9. *The k^{th} RAS iteration to solve Eq. (4.9) (with $M = I$) is a Dynamic Iteration as defined in Eq. (4.10) associated to the following splitting of the operators $A = A_1^d - A_2^d, B = B_1^d - B_2^d, C = C_1^a - C_2^a, D = D_1^a - D_2^a$:*

$$\begin{cases} \dot{x}^{(k)}(t) + A_1^d x^{(k)}(t) + B_1^d y^{(k)}(t) &= b^d(t) + A_2^d x^{(k-1)}(t) + B_2^d y^{(k-1)}(t), \\ C_1^a x^{(k)}(t) + D_1^a y^{(k)}(t) &= b^a(t) + C_2^a x^{(k-1)}(t) + D_2^a y^{(k-1)}(t), \\ x^{(k)}(0) = x_0, & t \in [0, T]. \end{cases} \quad (4.24)$$

with

$$\begin{aligned} A_1^d &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} A_i R_i^{p,d}, & A_2^d &= - \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} E_{i,d}^d R_{i,e}^{p,d}, & b^d(t) &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} R_i^{p,d} b^d(t), \\ B_1^d &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} B_i R_i^{p,a}, & B_2^d &= - \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} E_{i,d}^a R_{i,e}^{p,a}, \\ C_1^a &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} C_i R_i^{p,d}, & C_2^a &= - \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} E_{i,a}^d R_{i,e}^{p,d}, \\ D_1^a &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} D_i R_i^{p,a}, & D_2^a &= - \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} E_{i,a}^a R_{i,e}^{p,a}, & b^a(t) &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} R_i^{p,a} b^a(t). \end{aligned}$$

Proof. By applying the operator \tilde{R}_i^0 to Eq (4.21) and by summing the contribution of each partition W_i^p , we obtain:

$$\begin{aligned}
\sum_{i=0}^{N-1} \tilde{R}_i^{0,d} \dot{x}_i^{(k)}(t) + \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} A_i x_i^{(k)}(t) + \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} B_i y_i^{(k)}(t) &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} b_{i,d}(t) \\
&\quad - \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} E_{i,d}^d x_{ie}^{(k-1)}(t) - \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} E_{i,d}^a y_{ie}^{(k-1)}(t), \\
\sum_{i=0}^{N-1} \tilde{R}_i^{0,a} C_i x_i^{(k)}(t) + \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} D_i y_i^{(k)}(t) &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} b_{i,a}(t) \\
&\quad - \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} E_{i,a}^d x_{ie}^{(k-1)}(t) - \sum_{i=0}^{N-1} \tilde{R}_i^{0,a} E_{i,a}^a y_{ie}^{(k-1)}(t).
\end{aligned}$$

Hence, by using the definitions $x_i(t) = R_i^{p,d} x(t)$, $y_i(t) = R_i^{p,a} y(t)$, and the properties $\sum_{i=0}^{N-1} \tilde{R}_i^{0,d} \dot{x}(t) = \dot{x}(t)$, $\sum_{i=0}^{N-1} \tilde{R}_i^{0,d} b_i^d(t) = b^d(t)$ and $\sum_{i=0}^{N-1} \tilde{R}_i^{0,a} b_i^a(t) = b^a(t)$, we obtain Eq (4.24).

Finally, we need to show that $A = A_1^d - A_2^d$, $B = B_1^d - B_2^d$, $C = C_1^a - C_2^a$, $N = N_1^a - N_2^a$. By using the definition of $A_i = \tilde{R}_i^{p,d} A (R_i^{p,d})^T$ and $E_{i,d}^d = R_i^{p,d} A (R_{i,e}^{p,d})^T$, we have:

$$\begin{aligned}
A_1^d - A_2^d &= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} A_i R_i^{p,d} + \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} E_{i,d}^d R_{ie}^{p,d}, \\
&= \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} R_i^{p,d} A ((R_i^{p,d})^T R_i^{p,d} + (R_{i,e}^{p,d})^T R_{ie}^{p,d})
\end{aligned}$$

The definitions of $R_i^{p,d}$ and $R_{i,e}^{p,d}$ make that $R_i^{p,d} A ((R_i^{p,d})^T R_i^{p,d} + (R_{i,e}^{p,d})^T R_{ie}^{p,d}) = R_i^{p,d} A$. So, we have:

$$A_1^d - A_2^d = \sum_{i=0}^{N-1} \tilde{R}_i^{0,d} R_i^{p,d} A = A$$

The same calculation is done for $B_1^b - B_2^d = B$ and the property $R_i^{p,a} C ((R_i^{p,a})^T R_i^{p,a} + (R_{i,e}^{p,a})^T R_{ie}^{p,a}) = R_i^{p,a} C$ leads to $C_1^a - C_2^a = C$ and so on for $D_1^a - D_2^a = D$. \square

Thus the RAS method applied to DAE system belongs to the Dynamic Iteration methods with a specific splitting of the operators. Therefore, we can apply the Theorem 1 if the matrix D_1^a is invertible (which implies that the DAE system 2.17 is a DAE system of index 1) showing the pure linear convergence or divergence of the DI with RAS splitting in the continuous case. The peculiarity of the DI with RAS splitting compared to the DI with a general splitting of Eq. (4.10) is that we only work on the interfaces $W_{i,e}^p$.

By noting $W_{i,e}^{p,d}$ and $W_{i,e}^{p,a}$ the differential and algebraical components of $W_{i,e}^p$. Then we can define $\Gamma^d = \{W_{0,e}^{p,d}, \dots, W_{N-1,e}^{p,d}\}$, $\Gamma^a = \{W_{0,e}^{p,a}, \dots, W_{N-1,e}^{p,a}\}$ and $\Gamma = \{\Gamma^d, \Gamma^a\}$ and R_Γ the restriction to the global interface $R_\Gamma = \begin{pmatrix} R_\Gamma^d & 0 \\ 0 & R_\Gamma^a \end{pmatrix}$ with $R_\Gamma^d = (R_{0,ie}^{p,d}, \dots, R_{N-1,ie}^{p,d})^T$, $R_\Gamma^a = (R_{0,ie}^{p,a}, \dots, R_{N-1,ie}^{p,a})^T$. Then, we can extend the Theorem 1 for the DI with RAS splitting, which extends the proposition 2 of chapter 2 to the continuous case (i.e. $\forall t \in]0, T[$).

Proposition 10. *The Dynamic Iteration with RAS splitting defined by Eq. (4.24) applied to a linear DAE system with D_1^a invertible has an error operator $P_{t,\Gamma}$ which does not depend on the iteration number, such that the restriction of the iteration to the global interface (i.e. $z_\Gamma^{(k)} = R_\Gamma z_\Gamma^{(k)}$) satisfies:*

$$z_\Gamma^{(k)} = P_{t,\Gamma} z_\Gamma^{(k-1)} + c \quad (4.25)$$

The interest of Eq. (4.25) is to show the pure linear convergence of the DI and the possibility of accelerating the convergence to the true solution $z^{(\infty)}$ with the Aitken's technique for accelerating convergence, if 1 is not an eigenvalue of $P_{t,\Gamma}$, as follows:

$$z_\Gamma^{(\infty)} = (I - P_{t,\Gamma})^{-1} (z_\Gamma^{(1)} + P_{t,\Gamma} z_\Gamma^{(0)}) \quad (4.26)$$

The $P_{t,\Gamma}$ error operator is valid over the entire time interval of interest $t \in]0, T[$.

Remark 13. *Let us notice that the eigenvalues of $P_{t,\Gamma}$ are continuous in t . Consequently, there may exist a $t^* \in]0, T[$ such that $\rho(P_{t^*-, \Gamma}) < 1$ and $\rho(P_{t^*+, \Gamma}) > 1$, i.e the DI method converges on a time interval of limited size $]0, t^*[-$ and then diverges. Nevertheless, thanks to the Aitken's convergence acceleration technique, we do not need to worry about the convergence or the divergence, we just need to not choose the time interval $]0, t^*]$.*

In the next section, we are going to consider the discrete version of the DI, i.e. we discretize the time derivative with an approximation scheme that can be equivalent to applying the continuous DI method over a time interval $]0, \Delta t[$ repeatedly, starting with an initial condition consisting of the converged solution of the previous time step or starting with an initial condition consisting of the iterated solution of the previous time step.

4.3.2 Discrete counterpart of the DI with RAS splitting

The discrete counterpart of the DI with RAS splitting is obtained by using a backward Euler for time discretization, other backward differences formula (BDF) schemes would give similar results with more complicated formula.

The discrete DI with RAS splitting using backward Euler discretizing is written:

$$\begin{cases} \tilde{A}_1^d x^{n+1,(k+1)} + \tilde{B}_1^d y^{n+1,(k+1)} &= \tilde{b}^{n+1,d} + \tilde{A}_2^d x^{n+1,(k)} + \tilde{B}_2^d y^{n+1,(k)}, \\ C_1^a x^{n+1,(k+1)} + D_1^a y^{n+1,(k+1)} &= b^{n+1,a} + C_2^a x^{n+1,(k)} + D_2^a y^{n+1,(k)}, \\ x^{0,(k+1)} &= x_0. \end{cases} \quad (4.27)$$

with $\tilde{A}_1^d = I_{n,1}^d + \Delta t A_1^d$, $\tilde{B}_1^d = \Delta t B_1^d$, $\tilde{A}_2^d = \Delta t A_2^d$, $\tilde{B}_2^d = \Delta t B_2^d$, $\tilde{b}^{n+1,d} = x^{n,*} + \Delta t b^{n+1,d}$ where $x^{n,*}$ will be defined later.

Locally, it is written with $x_i^{0,(k+1)} = R_i^{p,d} x_0$:

$$\underbrace{\begin{pmatrix} x_i^{n+1,(k+1)} \\ y_i^{n+1,(k+1)} \end{pmatrix}}_{z_i^{n+1,(k+1)}} = \underbrace{\begin{pmatrix} \tilde{A}_i & \tilde{B}_i \\ C_i & D_i \end{pmatrix}^{-1}}_{\tilde{\mathbb{A}}_i^{-1}} \underbrace{\begin{pmatrix} \tilde{b}_{i,d}^{n+1} \\ b_{i,a}^{n+1} \end{pmatrix}}_{\tilde{b}_i^{n+1}} - \underbrace{\begin{pmatrix} \tilde{E}_{i,d}^d & \tilde{E}_{i,d}^a \\ E_{i,a}^d & E_{i,a}^a \end{pmatrix}}_{\tilde{\mathbb{E}}_i} \underbrace{\begin{pmatrix} x_{i,e}^{n+1,(k)} \\ y_{i,e}^{n+1,(k)} \end{pmatrix}}_{z_{i,e}^{n+1,(k)}} \quad (4.28)$$

We have recovered the formulation of the RAS method from the chapter 2, nevertheless putting it back in the framework of the dynamic iteration method allows us to extend the method when it is applied to time evolution problems. We have two possibilities for the term $x^{n,*}$ with $x^{n,*} = x^{n,(k+1)}$ or $x^{n,*} = x^{n,(\infty)}$, which will have an impact on the Aitken acceleration of convergence. This choice will lead to two implementation strategies described in the next section.

4.4 Time stepping strategies for DI

This section describes the implementation of the DI with RAS splitting splitting accelerated by the Aitken technique for convergence acceleration, applied over a time interval $[t^0, t^F]$ with a constant time step Δt satisfying $t^F - t^0 = \Xi \Delta t$ with $\Xi \in \mathbb{N}^*$.

4.4.1 Sequential DI strategy

The sequential DI strategy consists in iterating the DI method until convergence on one time step before applying it to the next time step. This strategy corresponds to the one which was applied in the two previous chapters and corresponds to the choice $x^{n,*} = x^{n,(\infty)}$ in Eq. (4.27).

In this sequential DI strategy, we can apply the Aitken's technique for accelerating convergence, after $n_\Gamma + 1$ DI iterations for the first regular time step, in order to numerically build the P operator. Then, if we use the same time step size for the following time steps, and if there is no non-linearity and no change in the topology, we can perform the Aitken's convergence acceleration technique after one DI iteration. The algorithm 3 describes the sequential DI strategy.

Algorithm 3 Sequential DI strategy

-
- 1: $n=1$ (first time step)
 - 2: Perform $n_\Gamma + 1$ DI with RAS splitting with algorithm 2 to obtain $[z_\Gamma^{1,(0)}, \dots, z_\Gamma^{1,(n_\Gamma+1)}]$
 - 3: Compute $P_{\Delta t, \Gamma} = [e_\Gamma^{(2)}, \dots, e_\Gamma^{(n_\Gamma+1)}][e_\Gamma^{(1)}, \dots, e_\Gamma^{(n_\Gamma)}]^{-1}$ with $e_\Gamma^{(i)} = z_\Gamma^{1,(i)} - z_\Gamma^{1,(i-1)}$.
 - 4: Perform the Aitken's formula to obtain:

$$z_\Gamma^{1,(\infty)} = (I - P_{\Delta t, \Gamma})^{-1}(z_\Gamma^{1,(n_\Gamma+1)} - P_{\Delta t, \Gamma} z_\Gamma^{1,(n_\Gamma)})$$

followed by one local solve to obtain $z^{1,(\infty)}$

- 5: **for** $n = 1$ to $\Xi - 1$ **do**
- 6: Perform 1 DI with RAS splitting with algorithm 2 starting from the initial guess $z_\Gamma^{n,(0)} = z_\Gamma^{n-1,(\infty)}$ to obtain $z_\Gamma^{n,(1)}$
- 7: Perform the Aitken's formula to obtain:

$$z_\Gamma^{n,(\infty)} = (I - P_{\Delta t, \Gamma})^{-1}(z_\Gamma^{n,(1)} - P_{\Delta t, \Gamma} z_\Gamma^{n,(0)})$$

followed by one local solve to obtain $z^{n,(\infty)}$

- 8: **end for**
-

4.4.2 Pipelined DI strategy

In the pipelined DI strategy, each DI iteration is performed over several time steps, these iterations are repeated until convergence. This strategy corresponds to the choice $x^{n,*} = x^{n,(k)}$ in Eq. (4.27).

We adopt the convention $z^{0,(k)} = z^0, \forall k$, The algorithm 4 summarizes the pipelined DI strategy (without Aitken's acceleration technique) for a period from t^0 to t^F with $t^F - t^0 = \Xi \Delta t$. There are Ξ consecutive time steps that are pipelined.

Algorithm 4 pipelined DI strategy without acceleration

-
- 1: **for** $k = 1 \dots$ until convergence **do**
 - 2: **for** $n = 0 \dots \Xi - 1$ **do**
 - 3: Perform one DI with RAS splitting with algorithm 2 starting with initial condition $z^{n,(k-1)}$ for the time step n .
 - 4: **end for**
 - 5: **end for**
-

Remark 14. *By comparing the sequential algorithm 3 and the pipelined algorithm 4, we can highlight two differences, the most obvious is the difference in the equations to be solved. However, there is also a difference in the order of the time loop*

and that of Schwarz. Indeed, in the sequential case the Schwarz loop is included in the time loop whereas in the pipelined case it is the other way around.

We want to perform an Aitken's acceleration of the convergence also with the pipelined strategy. For this purpose, we are going to write the pipelined DI iteration as a sequential DI iteration putting the iterated solutions on time steps in a vector form.

Definition 13. We note $Z_i^{(k)} \in \mathbb{C}^{\Xi n}$ the $(k)^{th}$ DI iteration corresponding to the concatenation over the Ξ time steps of the i^{th} partition W_i^p of the $(k+1)^{th}$ pipelined DI iteration:

$$Z_i^{(k)} = ((z_i^{1,(k)})^T, \dots, (z_i^{\Xi,(k)})^T)^T \quad (4.29)$$

Similarly we define $Z_{i,e}^{(k)}$ such that:

$$Z_{i,e}^{(k)} = ((z_{i,e}^{1,(k)})^T, \dots, (z_{i,e}^{\Xi,(k)})^T)^T \quad (4.30)$$

We also define the operator $\mathbb{I}_{d,i}$ such that:

$$\mathbb{I}_{d,i} z_i^{n,(k)} = \begin{pmatrix} \Delta t x_i^{n,(k)} \\ 0_{n_{i,a}} \end{pmatrix} \quad (4.31)$$

Proposition 11. With $Z_i^{(k)}$ defined by Eq. (4.29), the k^{th} pipelined DI iteration applied on to Ξ time steps Δt is written locally on partition W_i^p :

$$\mathcal{A}_i Z_i^{(k)} = \mathcal{B}_i - \mathcal{E}_i Z_{i,e}^{(k-1)} \quad (4.32)$$

with

$$\mathcal{A}_i = \begin{pmatrix} \tilde{\mathbb{A}}_i & & & & \\ -\mathbb{I}_{d,i} & \tilde{\mathbb{A}}_i & & & \\ & & \ddots & \ddots & \\ & & & -\mathbb{I}_{d,i} & \tilde{\mathbb{A}}_i \end{pmatrix} \quad (4.33)$$

$$\mathcal{B}_i = \begin{pmatrix} b_i^1 + \mathbb{I}_{d,i} z_i^0 \\ b_i^2 \\ \vdots \\ b_i^\Xi \end{pmatrix} \quad (4.34)$$

$$\mathcal{E}_{i,e} = \begin{pmatrix} \tilde{\mathbb{E}}_i & & & \\ & \tilde{\mathbb{E}}_i & & \\ & & \ddots & \\ & & & \tilde{\mathbb{E}}_i \end{pmatrix} \quad (4.35)$$

Proof. By starting with the pipelined DI iteration k on the partition W_i^p on the n^{th} time step and using the definition of $x^{n-1,*} = x^{n-1,(k)}$, we have:

$$\underbrace{\begin{pmatrix} \tilde{A}_i & \tilde{B}_i \\ C_i & D_i \end{pmatrix}}_{\tilde{\mathbb{A}}_i} \underbrace{\begin{pmatrix} x_i^{n,(k)} \\ y_i^{n,(k)} \end{pmatrix}}_{z_i^{n,(k)}} = \underbrace{\begin{pmatrix} b_{i,d}^n + \Delta t x_i^{n-1,(k)} \\ b_{i,a}^n \end{pmatrix}}_{\tilde{b}_i^n} - \underbrace{\begin{pmatrix} \tilde{E}_{i,d}^d & \tilde{E}_{i,d}^a \\ E_{i,a}^d & E_{i,a}^a \end{pmatrix}}_{\tilde{\mathbb{E}}_i} \underbrace{\begin{pmatrix} x_{i,e}^{n,(k-1)} \\ y_{i,e}^{n,(k-1)} \end{pmatrix}}_{z_{i,e}^{n,(k-1)}} \\ - \mathbb{I}_{d,i} z_i^{n-1,(k)} + \tilde{\mathbb{A}}_i z_i^{n,(k)} = b_i^n - \tilde{\mathbb{E}}_i z_{i,e}^{n,(k-1)}, \quad n = 1, \dots, \Xi \quad (4.36)$$

by writing Eq. (4.36) in matrix form we obtain Eq. (4.32). \square

Remark 15. Equation (4.32) has the same form (and the same properties) than the sequential DI. We can then apply the same methodology as in the sequential case. That is to say: as the convergence is purely linear one can thus build an operator of error and use the Aitken acceleration of the convergence method.

We can then calculate an error operator \mathbb{P}_Γ of the pipelined DI either algebraically or numerically. It will be of size $\Xi n_\Gamma \times \Xi n_\Gamma$ and we will need Ξn_Γ RAS iterations to calculate it numerically. Nevertheless, we can take advantage of the structure of \mathbb{P}_Γ for linear DAE with regular time stepping as the \mathbb{P}_Γ operator and the P_Γ operator are linked as shown in the next subsection.

4.4.3 Link between the error operators of sequential DI and pipelined DI

The sequential DI and the pipelined DI are linked when we consider linear DAE problem with regular time stepping. We are going to show that it is possible to calculate the error operator P_Γ of the sequential DI for the first time step and then to use it to build the error operator \mathbb{P}_Γ corresponding to the pipelined strategy. As for the sequential DI, we can restrict the iteration to the global interface values on Γ of Eq. (4.26) of all the Ξ time steps.

Definition 14. Let $Z_\Gamma^{(k)} \in \mathbb{C}^{\Xi n_\Gamma}$ denote the k^{th} pipelined DI iterations of the global interface values of the Ξ time steps:

$$Z_\Gamma^{(k)} = ((z_\Gamma^{1,(k)})^T, \dots, (z_\Gamma^{\Xi,(k)})^T)^T \quad (4.37)$$

let \mathbb{I}_d be the operator that follows:

$$\mathbb{I}_d = (\mathbb{I}_{d,1}^T, \dots, \mathbb{I}_{d,\Xi}^T)^T \quad (4.38)$$

By noting $M_{n,RAS}^{-1}$ the RAS operator and $P_{n,\Gamma}$ the error operator associated to the n^{th} time step. Then we can restrict the pipelined DI iteration to the global interface of all the Ξ time steps:

Proposition 12. *The k^{th} iteration of the pipelined DI can write on the global interface of the Ξ time steps:*

$$Z_{\Gamma}^{(k)} = \mathbb{P}_{\Gamma} Z_{\Gamma}^{(k-1)} + C \quad (4.39)$$

Where

$$\mathbb{P}_{\Gamma} = \left(\begin{array}{cccc} I & & & \\ M_2^{-1} & I & & \\ & \ddots & \ddots & \\ & & & M_{\Xi}^{-1} & I \end{array} \right)^{-1} \left(\begin{array}{cccc} P_{1,\Gamma} & & & \\ & P_{2,\Gamma} & & \\ & & \ddots & \\ & & & P_{\Xi,\Gamma} \end{array} \right) \quad (4.40)$$

$$M_i^{-1} = R_{\Gamma} M_{i,RAS}^{-1} \mathbb{I}_d R_{\Gamma}^T, \quad i = 2 \dots \Xi \quad (4.41)$$

$$C = \left(\begin{array}{c} R_{\Gamma} M_{1,RAS}^{-1} \mathbb{I}_d z^0 + c_1 \\ c_2 \\ \vdots \\ c_m \end{array} \right) \quad (4.42)$$

Proof. By starting with the Eq. (2.25) on the time step n and the associated $M_{n,RAS}^{-1}$ RAS operator associated to the global matrix \mathbb{A}^n , we adapt with the choice of $x^{*,(k)}$ (i.e. $\mathbb{I}_d z^{n-1,(k-1)}$) to define the k^{th} pipelined DI iteration:

$$\begin{aligned} z^{n,(k)} &= z^{n,(k-1)} + M_{n,RAS}^{-1} (b^n + \mathbb{I}_d z^{n-1,(k)} - \mathbb{A}^n z^{n,(k-1)}) \\ z^{n,(k)} - M_{n,RAS}^{-1} \mathbb{I}_d z^{n-1,(k)} &= z^{n,(k-1)} + M_{n,RAS}^{-1} (b^n - \mathbb{A}^n z^{n,(k-1)}) \end{aligned}$$

By restricting to the Γ boundary the last equation and remembering that $P_{n,\Gamma} = R_{\Gamma} (I - M_{n,RAS}^{-1} \mathbb{A}^n) R_{\Gamma}^T$, it gives:

$$R_{\Gamma} (z^{n,(k)} - M_{n,RAS}^{-1} \mathbb{I}_d z^{n-1,(k)}) = P_{n,\Gamma} R_{\Gamma} z^{n,(k-1)} + R_{\Gamma} M_{n,RAS}^{-1} b^n$$

as we have:

$$M_{n,RAS}^{-1} \mathbb{I}_d z^{n-1,(k)} = M_{n,RAS}^{-1} \mathbb{I}_d R_{\Gamma}^T R_{\Gamma} z^{n-1,(k)}$$

we can write with $c_n = R_{\Gamma} M_{n,RAS}^{-1} b^n$:

$$\begin{aligned} (I - R_{\Gamma} M_{n,RAS}^{-1} \mathbb{I}_d R_{\Gamma}^T) z_{\Gamma}^{n,(k)} &= P_{n,\Gamma} z_{\Gamma}^{n,(k-1)} + c_n \\ (I - M_n^{-1}) z_{\Gamma}^{n,(k)} &= P_{n,\Gamma} z_{\Gamma}^{n,(k-1)} + c_n \end{aligned}$$

Putting the last equation in matrix form gives the result. \square

The proposition 12 allows to optimize the pipelined DI in the case we solved a linear DAE with regular time stepping as then $P_{i,\Gamma} = P_{1,\Gamma}, \forall i$ and $M_{i,\Gamma}^{-1} = M_{1,\Gamma}^{-1}, \forall i$.

Consequently, only $n_\Gamma + 1$ sequential DI on the first step are needed to obtain $P_{1,\Gamma}$ numerically and then we can construct the \mathbb{P}_Γ error operator of the pipelined DI. Only one iteration of the pipelined DI is needed to obtain the interface solution on all time steps, then a local resolution on each time step gives the solution. Algorithm 5 describes this pipelined DI strategy with acceleration for linear DAE with regular time stepping:

Algorithm 5 pipelined DI strategy with acceleration for linear DAE with regular time stepping

- 1: $n=1$ (first time step)
- 2: Perform $n_\Gamma + 1$ DI with RAS splitting with algorithm 2 to obtain $[z_\Gamma^{1,(0)}, \dots, z_\Gamma^{1,(n_\Gamma+1)}]$
- 3: Compute $P_{1,\Gamma} = [e_\Gamma^{(2)}, \dots, e_\Gamma^{(n_\Gamma+1)}][e_\Gamma^{(1)}, \dots, e_\Gamma^{(n_\Gamma)}]^{-1}$ with $e_\Gamma^{(i)} = z_\Gamma^{1,(i)} - z_\Gamma^{1,(i-1)}$.
- 4: Compute \mathbb{P}_Γ from M_1^{-1} and $P_{1,\Gamma}$
- 5: $k = 1$ perform **one** pipelined DI iterate over the Ξ time steps.
- 6: Perform the Aitken's formula to obtain:

$$Z_\Gamma^{(\infty)} = (I - \mathbb{P}_\Gamma)^{-1}(Z_\Gamma^{(1)} - \mathbb{P}_\Gamma Z_\Gamma^{(0)})$$

followed by one local solve on each time step to obtain $Z^{(\infty)}$.

4.4.4 Comparison between Sequential and pipelined strategy

In this subsection we study the advantage and disadvantage of each of the two strategies depending on the situation, and more specifically the number of iterations needed to use the Aitken convergence acceleration method.

The error operator can be calculated in two ways algebraically Eq. 2.30 or numerically Eq. 3.9. We recall that the global interface Γ is defined as the concatenation of $W_{i,e}^p$, that is $\Gamma = \{W_{0,e}^p, \dots, W_{N-1,e}^p\}$ of size $n_\Gamma = \sum_{i=0}^{N-1} n_{i,e}$. It is pointed out that to numerically calculate the error operator, it is necessary to perform one more iteration than the size of the vector to be accelerated.

Consider the case of a period $\Xi\Delta t$ during which there will be no change in network topology, no nonlinearity and no change in time step size:

- sequential strategy: It would take $n_\Gamma + 1$ iterations to calculate the error operator, and then P_Γ would be the same for each time step and therefore it would suffice to perform one DI iteration per time step. So to compute the simulation over the whole period, we would need $\Xi + n_\Gamma$ iterations.

- first sequential time step and the rest with the pipelined strategy: it would still take $n_\Gamma + 1$ to calculate the error operator P_Γ , then deduce \mathbb{P}_Γ . Then, since we don't have the previous converged time step, we would need two DI iterations of the whole period. So to compute the whole simulation we would need $2(\Xi - 1) + n_\Gamma + 1$ iterations.
- pipelined strategy: it would take $\Xi \times n_\Gamma + 1$ to numerically compute the error operator \mathbb{P}_Γ and accelerate convergence to the true solution.

Now consider the case of a period $\Xi\Delta t$ during which there will be j changes in network topology, or j nonlinearities or j changes in time step size. It is necessary to recalculate the error operator P_Γ at each change of topology or at each change of time step. Indeed the matrices \mathbb{A} and \mathbb{E} which have an impact in the P_Γ are modified with the changes of topology or with the change of time step (because of the discretization). Let's compute the number of iterations needed for each strategy:











- sequential strategy: it would take $n_\Gamma + 1$ iterations to compute the error operator P_Γ each time needed (j occurrences), then it would be the same for each time step between changes and therefore it would suffice to carry out one DI iteration per time step during these periods. So to calculate the simulation over the whole period, it would take $\Xi - j + j \times (n_\Gamma + 1)$ iterations.
- first sequential time step after each change and the rest with the pipeline strategy: it would still take $n_\Gamma + 1$ to calculate the error operator P_Γ , then deduce \mathbb{P}_Γ after each change. Then, since we don't have the previous converged time step, we would need two DI iterations of the set of pipelined periods. So to compute the whole simulation we would need $2(\Xi - j) + j \times (n_\Gamma + 1)$ iterations.
- pipelined strategy: it would still take only $\Xi \times n_\Gamma + 1$ to numerically compute the error operator \mathbb{P}_Γ and accelerate convergence to the true solution.

Remark 16. *The strategy of computing the error operator using the sequential strategy and then computing the rest of the simulation with the pipelined strategy is never optimal. Indeed, it has the disadvantages of the other two strategies.*

Remark 17. *The pipelined strategy is a more planned strategy, in fact no rollback can be performed during the period $\Xi\Delta t$. Thus, for example, changes in the size of the time steps must be planned before launching the simulation, which is not the case in the sequential strategy.*

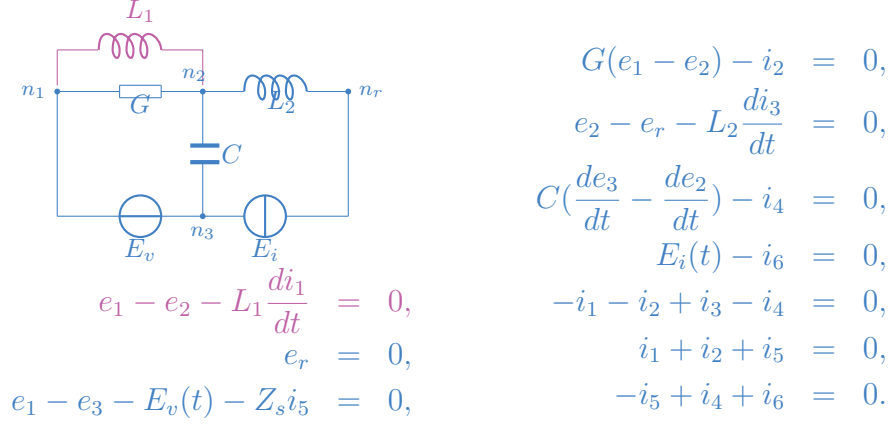
Remark 18. *The pipelined strategy is particularly useful in cases where there are nonlinear components in the network being studied. Indeed in this case we know before launching the simulation that there will be changes in the matrices at each time step. To carry out the simulation one linearizes with each step of time, the operator of error thus changes for the sequential case with each step of time. It will therefore take $\Xi \times (n_{\Gamma} + 1)$ for the simulation over the period $\Xi \Delta t$ with the sequential strategy while it will always be $\Xi \times n_{\Gamma} + 1$ for the pipelined strategy.*

Table 4.1: Summary of strategies

DI Strategy	Sequential	Pipelined
no non-linearity& fixed and equidistant time steps	 MORE EFFICIENT	 VALID
fixed variable time step distribution	 VALID	 MORE EFFICIENT
non-fixed variable time step distribution	 VALID	 NOT VALID
presence of non-linear components	 VALID	 MORE EFFICIENT
some rare non-linearity events	 MORE EFFICIENT	 VALID

4.5 Numerical Results on DI with RAS splitting strategies

The RLC circuit example was taken in [91], the circuit splitting is as follows:



In this example, the pink part needs the values e_1 and e_2 computed in the blue part, while the blue part needs the value i_1 calculated in the pink part.

4.5.1 Sequential strategy

In the sequential strategy, the error operator P_Γ and its eigenvalues are calculated following the method presented in section 2.4.1:

$$P = \begin{pmatrix} 0 & \begin{matrix} -\frac{\Delta t}{L_1} & \frac{\Delta t}{L_1} \end{matrix} \\ \begin{matrix} -\frac{L_2}{\Delta t} + \frac{\frac{L_2 C}{\Delta t^2} + \frac{L_2 G}{\Delta t} + 1}{\frac{C}{\Delta t} + G} \\ 0 \end{matrix} & \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix} \end{pmatrix}, \quad \begin{array}{c|c} \lambda_1 & 0 \\ \lambda_2 & -i\sqrt{\frac{\Delta t}{L_1(\frac{C}{\Delta t} + G)}} \\ \lambda_3 & i\sqrt{\frac{\Delta t}{L_1(\frac{C}{\Delta t} + G)}} \end{array}$$

The method diverges if $|\rho(P_\Gamma)| > 1$. L_1, C, G and L_2 are fixed so the convergence of the method depends on Δt . For $\Delta t_0 \stackrel{\text{def}}{=} \frac{L_1 G + \sqrt{(L_1 G)^2 + 4 L_1 C}}{2}$, we have $|\rho(P)| = 1$. The method converges with choosing a $\Delta t \in]0; \Delta t_0[$, stagnates if $\Delta t = \Delta t_0$ and diverges otherwise.

Figure 4.1 (left) shows the convergence of the sequential DI with the RAS splitting method with respect to the iterations and according to the Δt for one time step. The stagnation of the algorithm is numerically confirmed for $\Delta t = \Delta t_0$ (black line). The blue curve shows the divergence of the method for $\Delta t > \Delta t_0$ while the red curve exhibits the slow convergence when $\Delta t < \Delta t_0$. For both cases the Aitken's acceleration of the convergence allows to find the true solution. Figure

4.1 (right) shows the comparison of the e_3 solution obtained with the sequential DI with the RAS splitting accelerated by the Aitken's acceleration of the convergence technique on the time intervalle $[0, 0.1]$ with $\Delta t = 1.210^{-3}$ (divergent case) and the monolithic solution reference.

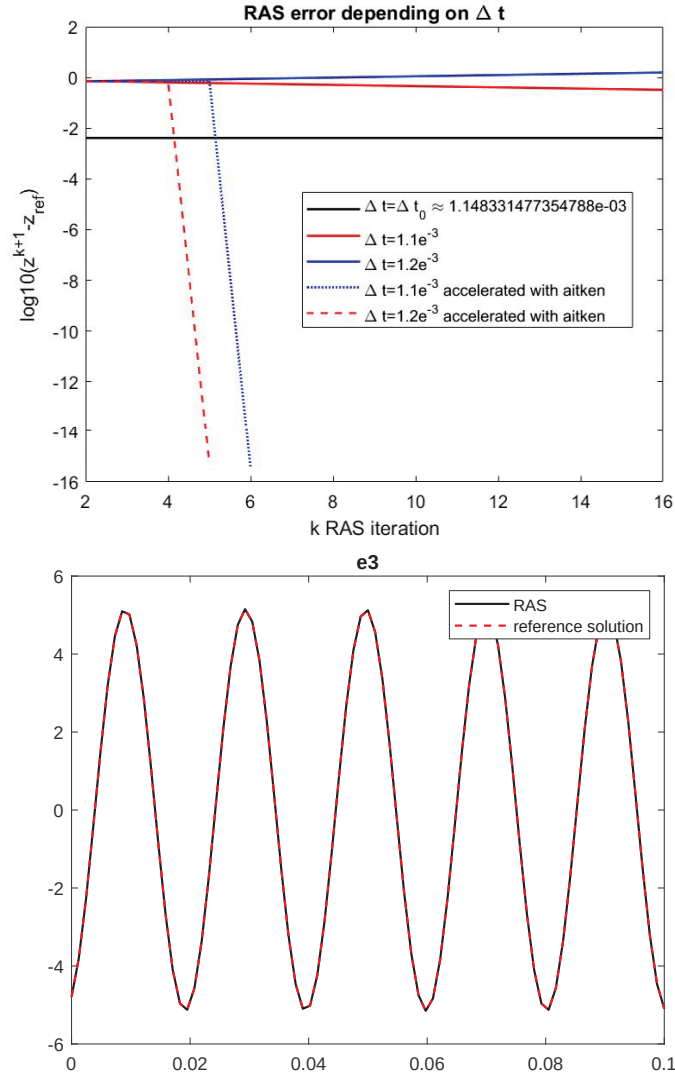


Figure 4.1: (top) Sequential DI with the RAS splitting convergence behavior with and without Aitken's acceleration ($\log_{10}(\|z^{(2k)} - z_{ref}\|_{\infty})$) with respect to the iterations for different values of Δt and one time step. (bottom) Comparison between the sequential DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference for the e_3 variable with $\Delta t = 1.2 \cdot 10^{-3}$, with parameters: $L_1 = 0.4$, $L_2 = 0.5$, $C = 1. \cdot 10^{-6}$, $G = 2. \cdot 10^{-3}$.

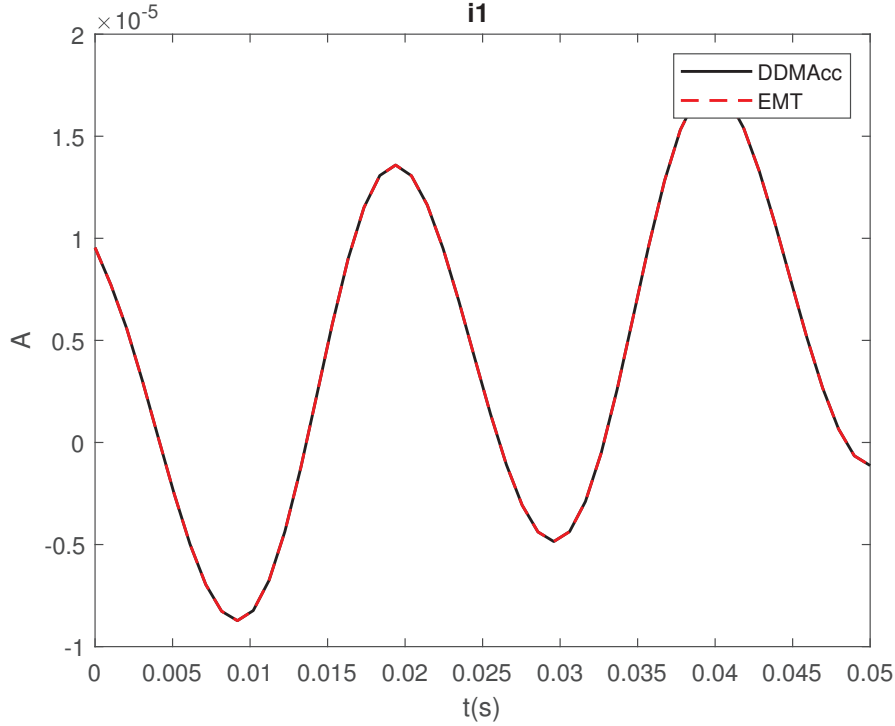


Figure 4.2: Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference for the i_1 variable computed with a sequential strategy with a non linear component, with $L_1 = 7$, $L_2 = 0.7$, $C = 1.10^{-6}$, $G_0 = 0.07$, $\alpha = 605$

The next numerical test considers a nonlinear component in the RLC circuit by taking the G component depending on the variable i_2 with $G = G_0 + \alpha i_2$. In this case, at each time step the system is linearized and P_Γ must be recalculated.

Figure 4.2 shows the evolution with respect to the time of i_1 of the RLC circuit with the nonlinear component G (with $\alpha = 605$ and $G_0 = 0.07$) for the monolithic computation and the sequential DI with RAS splitting. It exhibits that the sequential DI with RAS splitting find the monolithic solution reference.

4.5.2 Pipelined DI with RAS splitting strategy

Now we consider the pipelined DI with RAS splitting strategy applied to the RLC circuit. First we look on the effect on the convergence of the number of time steps involved in the pipeline. For this we compute numerically the \mathbb{P}_Γ operator for different number of pipelined time step and compute its spectral radius.

Figure 4.3 shows the evolution of the spectral radius of the error operator \mathbb{P}_Γ with a

time step $\Delta t = 1, 1 \cdot 10^{-3}$ on the left and $\Delta t = 1, 1 \cdot 10^{-4}$ on the right. It exhibits that the convergence of the pipelined DI with RAS splitting deteriorates with a number of pipelined time steps. For the regular time step $\Delta t = 1, 1 \cdot 10^{-3}$ the method is convergent up to 10 pipelined time steps and then diverges. Reducing the time step to $\Delta t = 1, 1 \cdot 10^{-4}$ allows to increase the number of pipelined time steps to 120 before the method diverges. The size of the time interval where the pipelined method can be applied then increases from $1, 1 \cdot 10^{-2}$ to $1, 32 \cdot 10^{-2}$. Nevertheless, Figure 4.3 (right) shows that after 100 pipelined time steps for $\Delta t = 1, 1 \cdot 10^{-4}$ the spectral radius increases less monotonously.

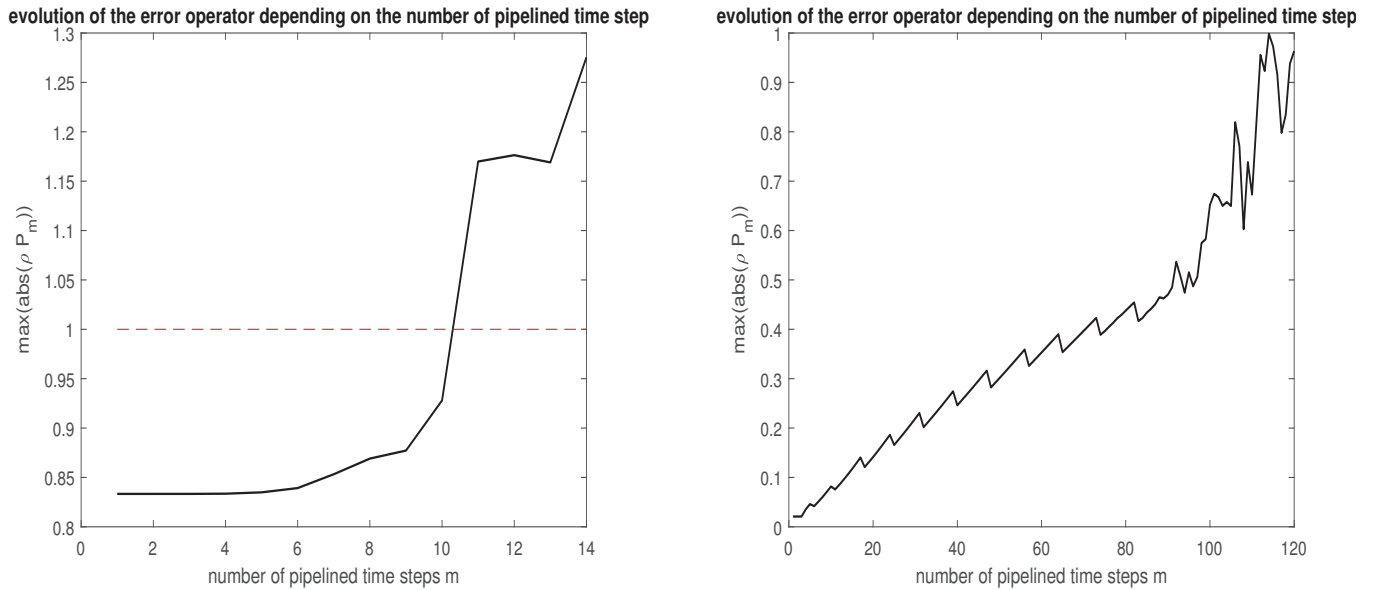


Figure 4.3: Evolution of the spectral radius of the error operator depending on the number of pipelined regular time steps of size $\Delta t = 1, 1 \cdot 10^{-3}$ (left, $\Xi = 14$) and $\Delta t = 1, 1 \cdot 10^{-4}$ (right, $\Xi = 120$), with $L_1 = 0.4$, $L_2 = 0.7$, $C = 1 \cdot 10^{-6}$, $G = 2 \cdot 10^{-3}$

Next we set the number of pipelined time step to be $\Xi = 14$ and we study how the error with the true solution and the pipelined DI with RAS splitting solution behaves over each of this Ξ time steps with respect to the number of iterations. Figure 4.4 (left), shows the pipelined DI with RAS splitting error with the true solution with respect to number of iterations. Each curve represents the error over one time step of the pipelined time steps. We are in the case where the spectral radius of \mathbb{P}_Γ is greater than one. Nevertheless, Figure 4.4 (left) exhibits that the error over time steps does not behave in the same way. The first two time steps are convergent while the others diverge. The more the time step is far from the first one, the more the divergence is great.

Although we are in the divergent case as we can compute numerically the \mathbb{P}_Γ

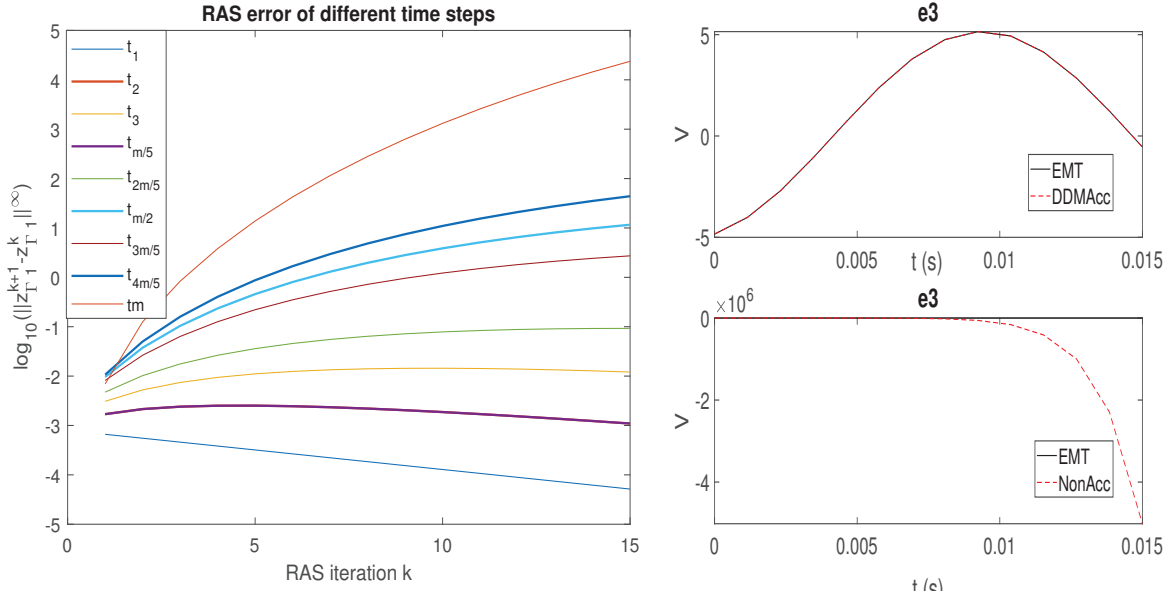


Figure 4.4: (Left) DI with the RAS splitting convergence behavior ($\log_{10}(\|z^{(2k)} - z_{ref}\|_{\infty})$) on each of the pipelined time steps with respect to the iterates and (right) comparison between the DI with the RAS splitting with and without the Aitken's technique for accelerating convergence and the DAE monolithic reference for the e_3 variable with $\Delta t = 1, 1.10^{-3}$, $\Xi = 14$, with parameters : $L_1 = 0.4$, $L_2 = 0.5$, $C = 1.10^{-6}$, $G = 2.10^{-3}$

operator associated to the Ξ pipelined time steps, we can perform the Aitken's acceleration of the convergence. Figure 4.4 (right), shows the behaviour of the e_3 on the time intervalle $[0, 0.015]$ for the pipelined DI with RAS splitting with (top) and without (bottom) Aitken's acceleration of the convergence technique and compares it to the monolithic solution. It exhibits that pipelined DI with RAS splitting fails to recover the true solution, unlike when acceleration is performed.

As mentioned in the comparison between the sequential and the pipelined DI with RAS splitting strategies, an advantage of the latter is that it can compute the \mathbb{P}_{Γ} associated with a problem with non-linear components and variable size time steps in the pipeline.

Figure 4.5 shows the results with the pipelined DI with RAS splitting strategy using Aitken's convergence acceleration method for the circuit problem with the nonlinear component $G = G_0 + \alpha i_2$. αi_2 is chosen very large compared to G_0 , which is not very physical but which makes it possible to appreciate the impact of the nonlinearity. We consider a pipeline of $\Xi = 100$ time steps of regular size

$\Delta t = 1.1 \cdot 10^{-4}$. The system is linearized at each time step, these changes are taken into account in the numerical computation of \mathbb{P}_Γ . The results exhibit a good agreement between the DI with RAS splitting solutions and the monolithic solution.

Figure 4.6 shows the results with the pipelined DI with RAS splitting using Aitken's convergence acceleration method where the time steps distribution in the pipeline changes twice: $\Delta t = 1.1 \cdot 10^{-3}$ on interval $[0, 0.008]$ (8 time steps), $\Delta t = 1.1 \cdot 10^{-4}$ on interval $[0.0081, 0.0101]$ (20 time steps) and $\Delta t = 1.1 \cdot 10^{-4}$ on interval $[0.0111, 0.03]$ (20 time steps) for a total of $\Xi = 48$ time steps in the pipeline. The \mathbb{P}_Γ error operator is computed numerically and the Aitken's acceleration of the convergence is then performed. Figure 4.6 exhibits the good agreement of the pipelined DI with RAS splitting accelerated solutions with the monolithic DAE solution.

4.6 Conclusion

In this chapter, the Dynamic Iteration method has been introduced. The DI method with RAS splitting as an iterative process involving the interface unknowns resulting from the partitioning of the system of differential algebraic equations has been formulated. It has thus been shown that the RAS method was part of a case of DI with a particular splitting making it possible to work only on the interfaces of the subdomains. Results from general DI can then be used to show the purely linear convergence of DI with RAS splitting in the continuous case. It was then shown that thanks to the method of acceleration of the convergence of Aitken one can get rid of the constraint of contractance of the error operator. Two strategies for using the DI scheme have been presented: a sequential strategy and a pipelined strategy. These strategies play on the use of macro time steps. The link between these two strategies was presented. Finally, the optimal use cases of these strategies were discussed. Numerical results show also the non dependence to the contraction of the error operator of the DI with RAS splitting accelerated by the Aitken's acceleration of the convergence technique. Moreover, the pipelined DI with RAS splitting succeeds to apply the Aitken's acceleration on nonlinear problem and pipelined steps with different sizes.

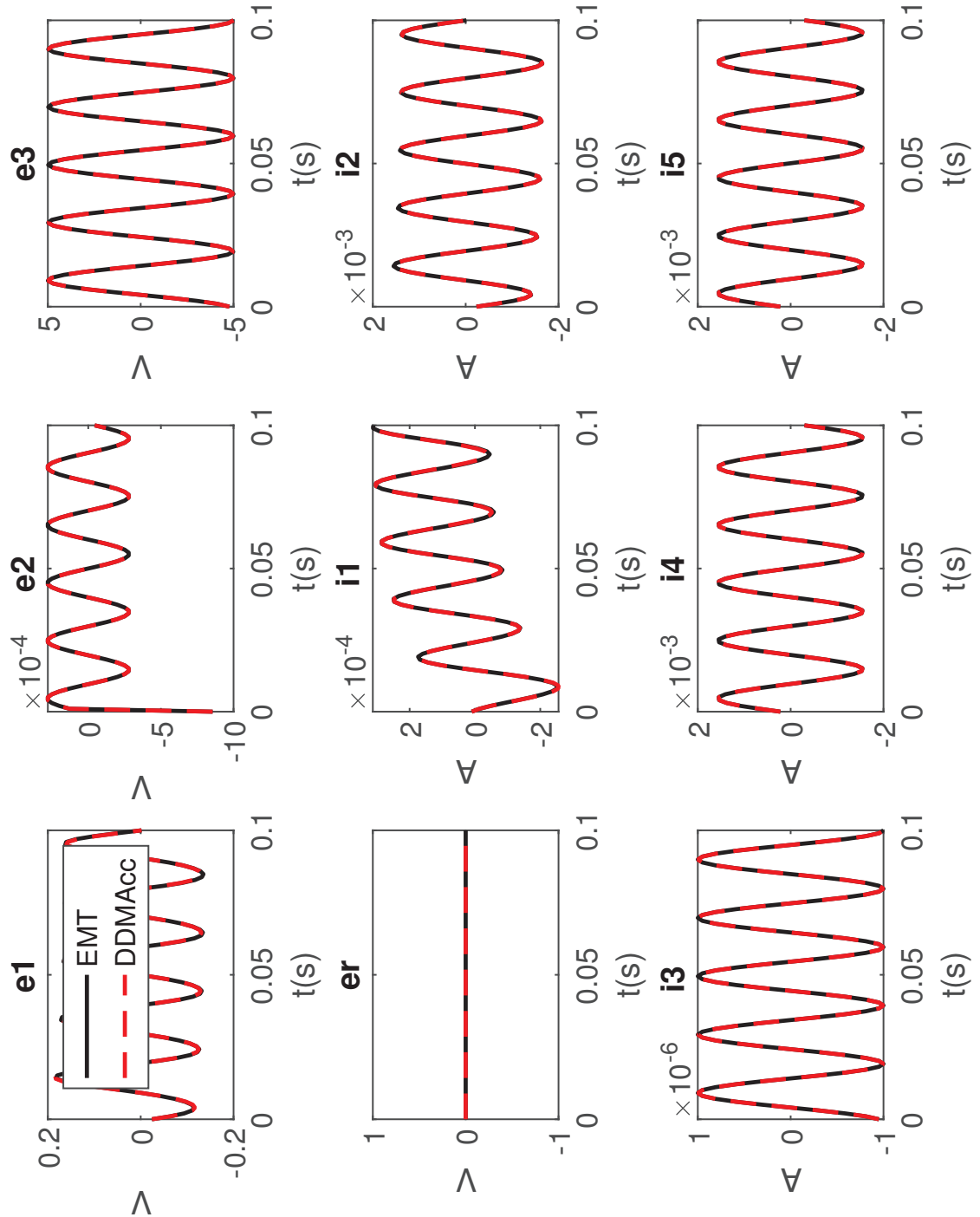


Figure 4.5: Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference with a non-linear component, with $\Delta t = 1, 1.10^{-4}$, $\Xi = 100$, with parameters: $L_1 = 2.7$, $L_2 = 0.9$, $C = 1.10^{-6}$, $G_0 = 0.07$, $\alpha = 605$

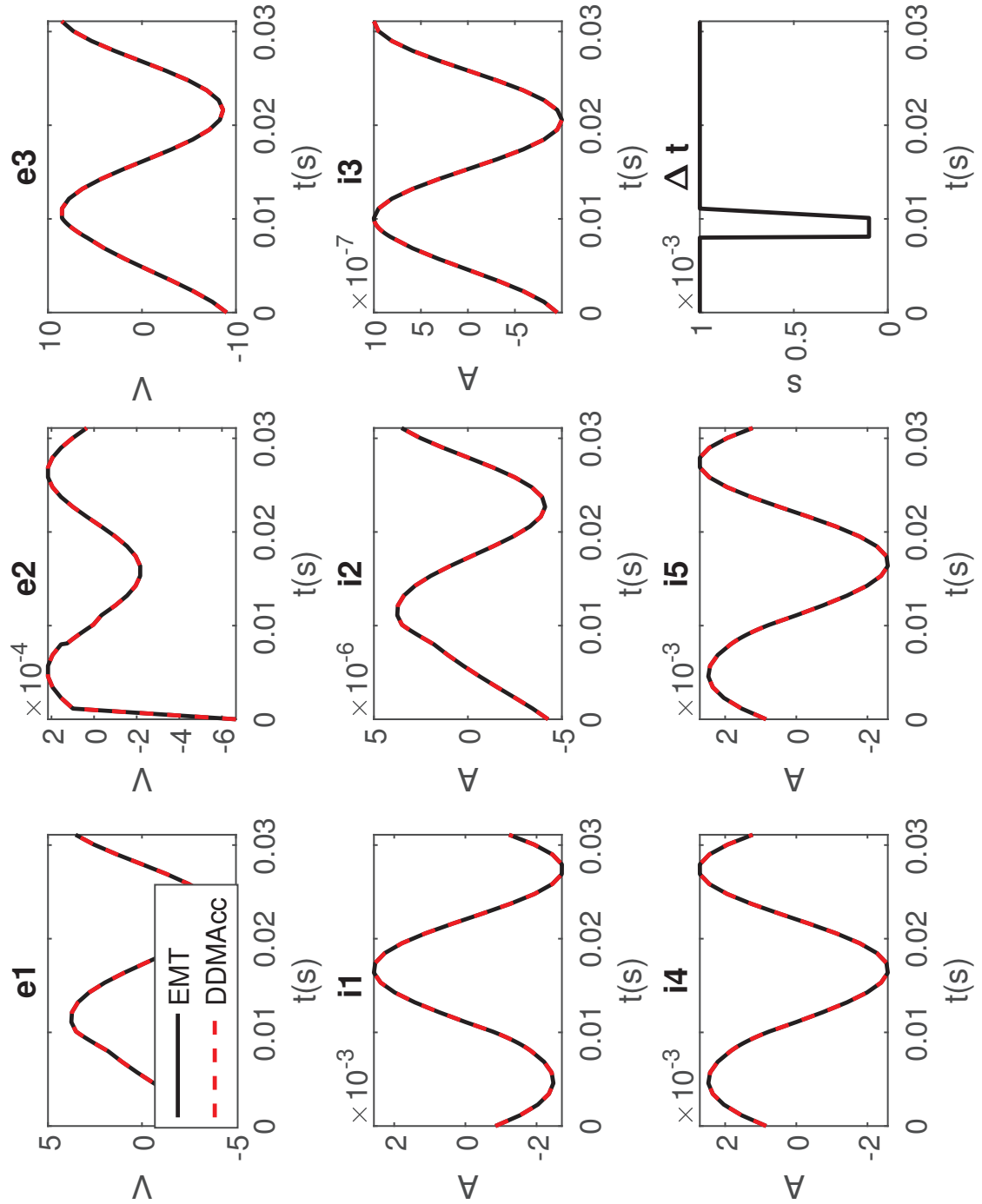


Figure 4.6: Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference with $\Delta t = 1, 1.10^{-3}$ from $t = 0$ to $t = 0.008$, $\Delta t = 1, 1.10^{-4}$ from $t = t = 0.0081$ to $t = 0.0101$ then $\Delta t = 1, 1.10^{-3}$ from $t = 0.0111$ to the end, $\Xi = 48$, with parameters: $L_1 = 5$, $L_2 = 0.7$, $C = 1.10^{-6}$, $G = 1.10^{-6}$

Chapter 5

Architecture

Contents

5.1	Introduction	107
5.2	OpenModelica	108
5.2.1	Modelica language	109
5.2.2	OpenModelica Advantages	110
5.2.3	OpenModelica disadvantages	113
5.2.4	Conclusion about OpenModelica	114
5.3	Functional Mockup Interface	115
5.3.1	OpenModelica FMU's export	116
5.4	Orchestrator	125
5.4.1	General Structure	126
5.4.2	FMU type slave	130
5.4.3	Conclusion About the Orchestrator	136

5.1 Introduction

One of the advantages of EMT-TS co-simulation is to reduce the computation time compared to an all-EMT simulation, for this reduction in computation time to have an impact, the circuit to be simulated must be large enough. We would therefore like to be able to simulate large electrical networks. Indeed the mathematical methods that we wish to use have been set up (in chapter 3) as well as the strategy

for using these methods (in chapter 4), however we still have studies to carry out which are only possible on major networks:

- to be able to compare the two representations on a substantial overlap in order to know the lost information and to rework on the passage of information,
- study the partitioning in more detail, thus making it possible to see up to what distance from a fault in the network, the impact of the fault remains relevant to study. As well as the size of the sub-domains so that their calculation weights are similar (despite the fact that they are not modeled in the same way) so that the parallelization is optimal,
- being able to confirm thanks to a larger overlap that the size of the overlap does not influence the convergence of the method.

To be able to use large networks, it was decided to use the OpenModelica tool. Indeed this tool being more and more used, we wanted to know its potential and its limits. This will have allowed us to know if we were going to generalize its use within the institute. One of the objectives of the thesis was therefore to develop skills and knowledge on the use of this tool. This chapter is organized as follows: First, we will present OpenModelica its advantages as well as its disadvantages. Then we will present another tool: FMI as well as its link with OpenModelica. Finally we will talk about the structure put in place to use these tools as well as the models used.

5.2 OpenModelica

OpenModelica is a suite of open source tools for modeling, simulation, and model-based development. It's development is supported by the the Open Source Modelica Consortium. This open source tool suite is composed among other things: the Open Modelica Compiler OMC, OMshell the modelica scripting tool and OMedit the OpenModelica Graphic Model Editor and Simulator GUI. OpenModelica is structured in several layers which make the models pass through the following stages: equation of the models; optimization; transition to an intermediate language: Meta Modelica; then generated in the target language, C or C++ then compiled by OMC. The OpenModelica tools are based on the Modelica language which we will present first. We will then discuss the interest of using OpenModelica, followed by the blocking points of the use of OpenModelica in our project and we will conclude on the exclusive use of this tool for our EMT TS Co-simulation.

5.2.1 Modelica language

Modelica is a declarative language used to describe mathematical behavior. Their main characteristics are explained by Fritzson in [38, 37]. This language is in constant evolution, the specifications are available on the site dedicated to Modelica [14]. The particularly interesting characteristics of this language, for the work presented in this thesis, are:

- Modelica is an object-oriented language that allows the reuse of template classes. Classes allow the modeling of components that can be reused in more complex models, providing hierarchical structuring. Additionally, Modelica defines class types, which allows libraries to be more structured. The most relevant of these types of classes are the connectors which make it possible to specify the exchanges of information between two Modelica classes, these connectors can have a physical representation like a pin in the electrical circuit.
- Modelica is an equation-based and acausal language. Acausality means that the order in which the equations are written has no impact and that the "=" operator is not an assignment but an equality. In this way, the components are described by their mathematical behavior. These components can then be combined causally or acausally to obtain complex physical systems.
- The Modelica language allows to mix continuous and discrete modeling. This enables to introduce discrete events into the simulation or to model an engineering system and its control in the same environment.
- Modelica allows the use of annotations to graphically represent systems.

This language is increasingly used in the electrical community, and some energy players are pushing for it to be used more and more as a unified language. Dymola and Dynawo [50]. are two examples of modeling and simulation software based on the Modelica language.

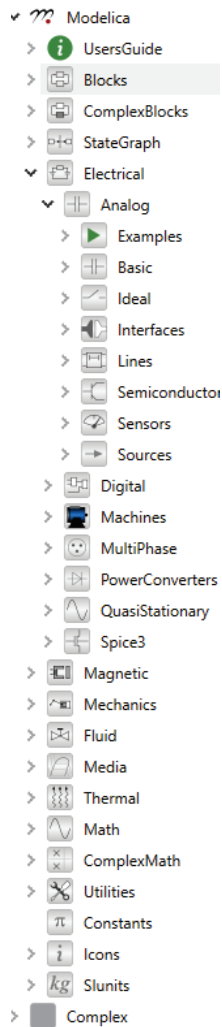


Figure 5.1: The Modelica Library

Several open-source libraries of electrical components are available, written in Modelica language:

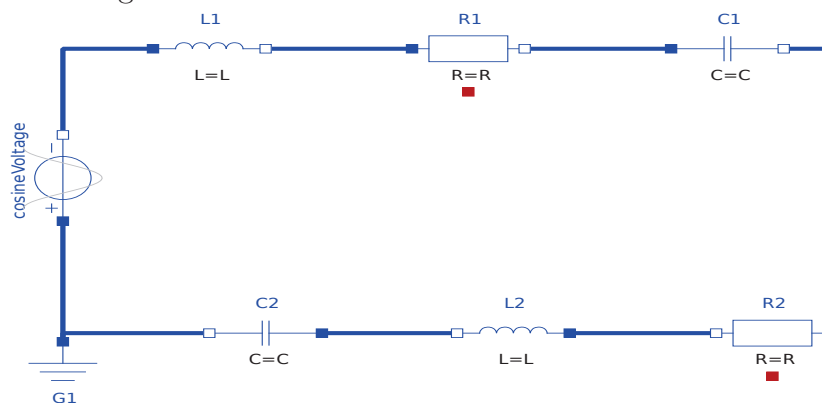
- The Modelica library: it is developed by the Modelica Association. It offers many components from many physical domains. The figure 5.1 is a screenshot of the electrical section of the Modelica library open in OMEdit.
- The OpenIpsl library (Open-Instance Power System Library): a library of power system component models developed by Luigi Vanfretti's research group ALSETLab at Rensselaer Polytechnic Institute [20].
- The ModPowerSystems library: a library which contains models of electrical systems developed in several different modeling types: static phasors, dynamic phasors as well as EMT. It is developed by the Institute of Automation of Complex Power System, in the E.ON Energy Research Center from RWTH University Aachen. [29].

5.2.2 OpenModelica Advantages

The advantages of using OpenModelica is first of all that it is free, which promotes collaboration between the different actors. Moreover, it is used more and more, we can cite for example a part of the Architecture program of SuperGrid Institute in which OpenModelica is used a lot, or the Dynawo tool developed within RTE which partly based on OpenModelica.

OMEdit has a user-friendly and intuitive graphical interface, a person who does not know how to code at all can click and drag the components under their graphical

representation, as can be seen in the figure 5.2 where we reconstruct the circuit of chapter 2 using click-and-drag components from the Modelica library (top image). The tool instantly converts the model thus created into a model written in Modelica language using the objects defined in the library (bottom image). Finally, figure 5.3 shows the instantiation of this class, during this instantiation we obtain the class with all the parameters and equations given to the same level. This instantiation has these advantages and disadvantages, the main advantage being that there is no additional work for the user, the main disadvantages already being that a very small circuit turns into a whole page of value and equations, the second drawback being that everything is put back to the same level in the class, which we will see later prevents doing a co-simulation in the tool.



```

model PetitCircuit
  Modelica.Electrical.Analog.Basic.Inductor L1 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Resistor R1 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Capacitor C1 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Resistor R2 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Inductor L2 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Capacitor C2 annotation(
    ...);
  Modelica.Electrical.Analog.Basic.Ground G1 annotation(
    ...);
  Modelica.Electrical.Analog.Sources.CosineVoltage cosineVoltage annotation(
    ...);
equation
  connect(cosineVoltage.p, G1.p) annotation(
    ...);
  connect(C2.n, G1.p) annotation(
    ...);
  connect(L2.n, C2.p) annotation(
    ...);
  connect(R2.n, L2.p) annotation(
    ...);
  connect(cosineVoltage.n, L1.p) annotation(
    ...);
  connect(L1.n, R1.p) annotation(
    ...);
  connect(R1.n, C1.p) annotation(
    ...);
  connect(C1.n, R2.p) annotation(
    ...);
annotation(
  ...);
end PetitCircuit;

```

Figure 5.2: Different representations of the same class "circuit" under OpenModelica


```

class PetitCircuit
  Real L1.v(quantity = "ElectricPotential", unit = "V") "Voltage drop of the two pins (= p.v - n.v)";
  Real L1.i(quantity = "ElectricCurrent", unit = "A", start = 0.0) "Current flowing from pin p to pin n";
  Real L1.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real L1.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real L1.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real L1.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  parameter Real L1.L(quantity = "Inductance", unit = "H", start = 1.0) "Inductance";
  Real R1.v(quantity = "ElectricPotential", unit = "V") "Voltage drop of the two pins (= p.v - n.v)";
  Real R1.i(quantity = "ElectricCurrent", unit = "A") "Current flowing from pin p to pin n";
  Real R1.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real R1.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real R1.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real R1.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  parameter Boolean R1.useHeatPort = false "=true, if heatPort is enabled";
  parameter Real R1.T(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15, nominal = 300.0);
  Real R1.LossPower(quantity = "Power", unit = "W") "Loss power leaving component via heatPort";
  Real R1.T_heatPort(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15, nominal = 300.0);
  parameter Real R1.R(quantity = "Resistance", unit = "Ohm", start = 1.0) "Resistance at temperature T_ref";
  parameter Real R1.T_ref(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15, nominal = 300.0);
  parameter Real R1.alpha(quantity = "LinearTemperatureCoefficient", unit = "1/K") = 0.0;
  Real R1.R_actual(quantity = "Resistance", unit = "Ohm") "Actual resistance = R*(1 + alpha*(T_heatPort - T_ref))";
  Real C1.v(quantity = "ElectricPotential", unit = "V", start = 0.0) "Voltage drop of the two pins (= p.v - n.v)";
  Real C1.i(quantity = "ElectricCurrent", unit = "A") "Current flowing from pin p to pin n";
  Real C1.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real C1.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real C1.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real C1.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";

  :
  :

  Real cosineVoltage.signalSource.y "Connector of Real output signal";
  parameter Real cosineVoltage.signalSource.offset = cosineVoltage.offset "Offset of output signal y";
  parameter Real cosineVoltage.signalSource.startTime(quantity = "Time", unit = "s") = cosineVoltage.startTime;
  parameter Real cosineVoltage.signalSource.amplitude = cosineVoltage.V "Amplitude of cosine wave";
  parameter Real cosineVoltage.signalSource.freqHz(quantity = "Frequency", unit = "Hz", start = 1.0) = cosineVoltage.freqHz;
  parameter Real cosineVoltage.signalSource.phase(quantity = "Angle", unit = "rad", displayUnit = "deg") = cosineVoltage.phase;
  parameter Real cosineVoltage.V(quantity = "ElectricPotential", unit = "V", start = 1.0) "Amplitude of cosine wave";
  parameter Real cosineVoltage.phase(quantity = "Angle", unit = "rad", displayUnit = "deg") = 0.0 "Phase of cosine wave";
  parameter Real cosineVoltage.freqHz(quantity = "Frequency", unit = "Hz", start = 1.0) "Frequency of cosine wave";

equation
  L1.L * der(L1.i) = L1.v;
  L1.v = L1.p.v - L1.n.v;
  0.0 = L1.p.i + L1.n.i;
  L1.i = L1.p.i;
  assert(1.0 + R1.alpha * (R1.T_heatPort - R1.T_ref) >= 1e-15, "Temperature outside scope of model!");
  R1.R_actual = R1.R * (1.0 + R1.alpha * (R1.T_heatPort - R1.T_ref));
  R1.v = R1.R_actual * R1.i;
  R1.LossPower = R1.v * R1.i;
  R1.v = R1.p.v - R1.n.v;
  0.0 = R1.p.i + R1.n.i;
  R1.i = R1.p.i;
  R1.T_heatPort = R1.T;
  C1.i = C1.C * der(C1.v);
  C1.v = C1.p.v - C1.n.v;

  :
  :

  cosineVoltage.signalSource.y = cosineVoltage.signalSource.offset + (if time < cosineVoltage.signalSource.startTime then 0.0
  else cosineVoltage.signalSource.amplitude * cos(6.283185307179586 * cosineVoltage.signalSource.freqHz *
  (time - cosineVoltage.signalSource.startTime) + cosineVoltage.signalSource.phase));
  cosineVoltage.v = cosineVoltage.signalSource.y;
  cosineVoltage.v = cosineVoltage.p.v - cosineVoltage.n.v;
  0.0 = cosineVoltage.p.i + cosineVoltage.n.i;
  cosineVoltage.i = cosineVoltage.p.i;
  L1.p.i + cosineVoltage.n.i = 0.0;
  L1.n.i + R1.p.i = 0.0;
  R1.n.i + C1.p.i = 0.0;
  C1.n.i + R2.p.i = 0.0;
  R2.n.i + L2.p.i = 0.0;
  L2.n.i + C2.p.i = 0.0;
  C2.n.i + G1.p.i + cosineVoltage.p.i = 0.0;
  C2.n.v = G1.p.v;
  C2.n.v = cosineVoltage.p.v;
  C2.p.v = L2.n.v;
  L2.p.v = R2.n.v;
  L1.p.v = cosineVoltage.n.v;
  L1.n.v = R1.p.v;
  C1.p.v = R1.n.v;
  C1.n.v = R2.p.v;
end PetitCircuit;

```

Figure 5.3: Instantiation of the class "PetitCircuit" on OpenModelica, a part has been cut, the complete instantiation is composed of 131 lines

Another advantage of OpenModelica are the flags that allow one to specify in the model the methods or modes to use to simulate this model. For example, there is a DAE mode and different index reduction methods for DAE systems. The goal being to reduce the system to an index 1, so that it can be simulated with common solvers. A simulation without index reduction is also possible, but less reliable. Several different definitions exist for the index of a DAE system, the most common definition is: The differential index of a system of algebraic differential equations is defined as the maximum number of differentiations of all equations such that all unknowns of the system can be solved by integrating only ordinary differential equations. In the case of electrical circuits, the index of the DAE system representing this electrical circuit is correlated to the topology of the circuit. Indeed, an inductor-like component connected to a circuit containing a loop with only capacitors or voltage sources will have index 2, or an inductor-like component connected to a circuit having a cutset with only inductors and current sources will be of index 2. Cortes Garcia & al demonstrated the correlation between the index of a DAE system and the circuit topology in [25]. This type of configuration is rare in the world of electricity, however they must be able to be taken into account in the simulation.

Thus several methods are proposed in OpenModelica such as the index reduction algorithm with dummy-states, described by Söderlind and Mattsson [76].

5.2.3 OpenModelica disadvantages

The first point to mention is that despite the efforts made to diversify the methods, the reduction of the DAE index regularly generates errors or failures. The calculations of the number of equations and variables are not intuitive and it is sometimes necessary to add "ghost" equations to pass the first check. These points are points which can delay the implementation of the co-simulation but which are not necessarily blocking because they can be circumvented. On the other hand, there are certain points which prevent us from setting up our co-simulation on the OpenModelica tool. These three main problems being:

- When instantiating a model, all equations and variables are put on the same level, to work around this we tried to make point connections between sub-models using the modelica language's ability to mix continuous time and discrete time.
- The connection between two subsets cannot be punctual, indeed a keyword "connect" results in a continuous connection when we want to fix certain variables on the edges of our sub-models. Without using a connector type class and the "connect" keyword, and using discrete events with the event

keyword "when", we get issues with the number of equations related to the number of variables or the variables are forced to zero.

- To implement the chosen method (see chapters 2 and 3), it is necessary to be able to replay the same time step several times. We can't manipulate time to our liking on OpenModelica, and among other things, we can't rollback; Indeed, in spite of the possibility of mixing discrete events with continuous simulation, one cannot restore the whole of the system such as it was at the beginning of the step of time. This fact definitely prevents us from applying a RAS method within OpenModelica. However, this does not prevent us from creating a hybrid by integrating a transformation of type DQ0 and inverse DQ0 in the sub-models. Which brings us to the last point that prevents our EMT-TS simulation.
- If we have an EMT model and a TS model (with proper translation like the DQ0 one) in OpenModelica and we connect them, the hybrid obtained is solved as a single whole model. Indeed this is again due to the fact that during the instantiation everything is put back to the same level. And therefore the time step used is the smallest necessary, that is to say, the one used to solve the EMT part. As a result, the advantage of the TS: its computational efficiency, is lost. We can therefore make a hybrid model on OpenModelica but which has less interest than a monolithic EMT simulation.

5.2.4 Conclusion about OpenModelica

OpenModelica is a tool with great potential, and user friendly, however it is not mature on a lot of points. The four blocking points mentioned above prevent us from implementing the RAS method and the EMT-TS Co-simulation in OpenModelica. Even if it is possible to create a hybrid under OpenModelica, we lose the advantages of the TS part. As the use of this tool is really one of the fixed objectives of this thesis, we want to explore all the avenues offered by this tool. We turned to the FMI, a tool that allows to use the models created under OpenModelica.

5.3 Functional Mockup Interface

The Functional Mockup Interface (FMI) is an open source tool developed and maintained as a Modelica Association project. It is an independent standard which defines an interface with functions allowing exchanges between several dynamic models.

This standard makes it possible to describe dynamic models using standardized XML files. These models are described in containers exported from the simulation environment as a compressed file. They are called Functional Mock-up Unit (FMU),

these FMUs contain: the model description in an xml file called modeldescription, binary files and C code files. Binaries are only compatible with one operating system, so FMUs can only be used on one operating system. This standardization is supported by more than 170 tools. See figure 5.4 which is part of the list of tools supporting this standardization. This list is taken directly from the FMI website.

Remark 19. *Although standardization in FMU is coded, it is not always possible to exchange models between tools. For example, a Dymola license is required on the machine on which an FMU exported from Dymola is used, unless an additional license has been paid for.*

Name	FMU Export				FMU Import			
	Co-Simulation		Model Exchange		Co-Simulation		Model Exchange	
Adams	1.0	2.0			1.0	2.0	1.0	2.0
Dymola	1.0	2.0	1.0	2.0	1.0	2.0	1.0	2.0
DS - FMU Ex...	1.0	2.0	1.0	2.0				
EMTP-RV		2.0			1.0	2.0		2.0
Easy5	1.0	2.0		2.0	1.0	2.0	1.0	2.0
General Energy Systems (GES)	1.0	2.0	1.0	2.0	1.0	2.0	1.0	2.0
JModelica.org	1.0	2.0	1.0	2.0	1.0	2.0	1.0	
MATLAB® Simulink®		2.0			1.0	2.0	1.0	2.0
OpenModelica		2.0	1.0	2.0		2.0	1.0	2.0
Simcenter Amesim	1.0	2.0	1.0	2.0	1.0	2.0	1.0	2.0
SystemModeler	1.0	2.0	1.0	2.0	1.0	2.0	1.0	2.0

Planned

Supported

Cross-Check passed

Figure 5.4: The modelica Library

Two types of FMU export are accessible, These two types of FMU have functions in common and additional functions to use the particularities of these two types of FMU, these two types are:

- Model Exchange (ME): These FMUs contain the description of the model dynamics but do not contain a solver. It is therefore necessary to provide an external solver which will allow each time step to solve the system, thanks to an approximation, calculated by the FMU, of the states and their derivatives.
- Co-Simulation (CS): These FMUs contain their own numerical solver (solver of the tool from which the FMU was exported). the user must define the inputs and outputs and manage the exchanges between the models.

Remark 20. *In OpenModelica you can export FMUs that are both Co-simulation and Model Exchange*

5.3.1 OpenModelica FMU's export

We will create our model under OpenModelica then export them in FMU format. We can notice in the 5.4 table, that the OpenModelica tool has no Cross-check passed for any of the 2 types of FMU, neither for export nor for import. Basing our simulations on it may therefore seem risky, however as knowing the state of the OpenModelica tool is one of the objectives of the thesis, we decided to deepen the subject as much as possible.

We have several Modelica models at SuperGrid Institute. For each of these circuits two equivalent copies were created: an EMT model and a TS model, they were created with the OpenModelica OMEdit tool. This allows us to be able to cut at will and create an overlap. The first thing to do is of course to cut the chosen model into a sub-model. Remember that OpenModelica is based on physics. When cutting a model, we end up with a circuit considered as open, it will therefore be necessary to add components which will allow both the recovery of information from another sub-domain, to select the information calculated in this sub-domain to send to other submodels and finally close the circuit.

5.3.1.1 Interface components

The Modelica library has two components used for this:

GeneralVoltageToCurrentAdaptor and *GeneralCurrentToVoltageAdaptor* which allow for the first to recover the voltage coming from another submodel and to transmit a calculated current in this sub- model, and for the second component it recover the currents of another sub-model and provide the voltage that the sub-model has calculated. Each of these interface components retrieves discrete data and inputs a continuous signal into the model via zero-order hold recomposition. We created our own components (suitable for our three-phase EMT models or our Phasor models) by taking the example of the two components present in the Modelica library. So we get two components suitable for EMT or TS. However,

these two components do not necessarily allow the creation of overlapping submodels. We therefore also create additional components for this kind of configuration, these components would be for example sensors and circuit closures. Indeed, an unconnected pin is perceived by the tool as an open circuit.

These different interface components must respect the following constraints:

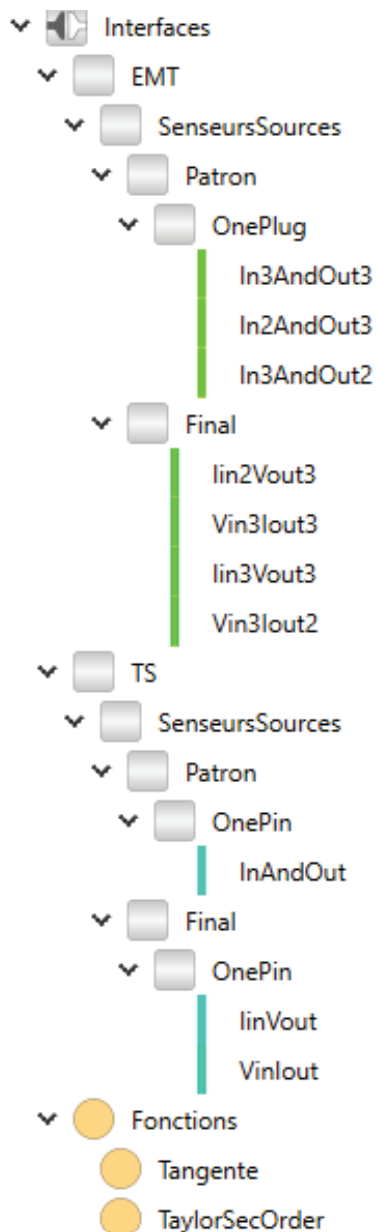


Figure 5.5: The modelica Library

- Respect for physics, indeed these fictitious components are perceived as components by OpenModelica but are not present in reality and should therefore not have any impact on the simulation. So, for example, current sensors must be able to be connected in series and voltage sensors in parallel with a current forced to zero.
- Reconstruction of an input signal for currents. Indeed, in Modelica the pins of the components are connectors whose current is annotated with the keyword "flow", in this way the sum of the currents of the pins connected between them is zero, in order to respect the law of the nodes. However, this property can cause problems because the inputs are not variables and therefore when deriving a value which is a combination of other streams and this input. This is problematic because this part cannot be derived by the tool, which creates an error.

Remark 21. *We did not have to face this problem in the previous chapters because we had previously discretized the system, and when an input was derived we had simply passed the values and that of the previous time step.*

For this constraint, there are functions in the Modelica library called *state1* and *state2* which take as input the value, the first derivative of the value (and the second derivative for *state2*) and return only the value. This way a signal is not actually reconstructed, but it does make the constant a variable that can be derived by the tool.

After creating these components, we tested them on OpenModelica. To do this, we cut the connections where we wanted to partition the model and placed the appropriate interface components there that we connected to each other. The interface components have been validated as having no impact in physics if the simulation results obtained are the same results as those obtained with the original model.

Figure 5.6 is obtained by exporting the graphical representation of a model from OMEdit. The model shown is a three bus model that we have in TS version and EMT version. it consists of 3 lines, an infinite Bus and a generator, the representation here is the TS version one. Interface components are tested in this model.

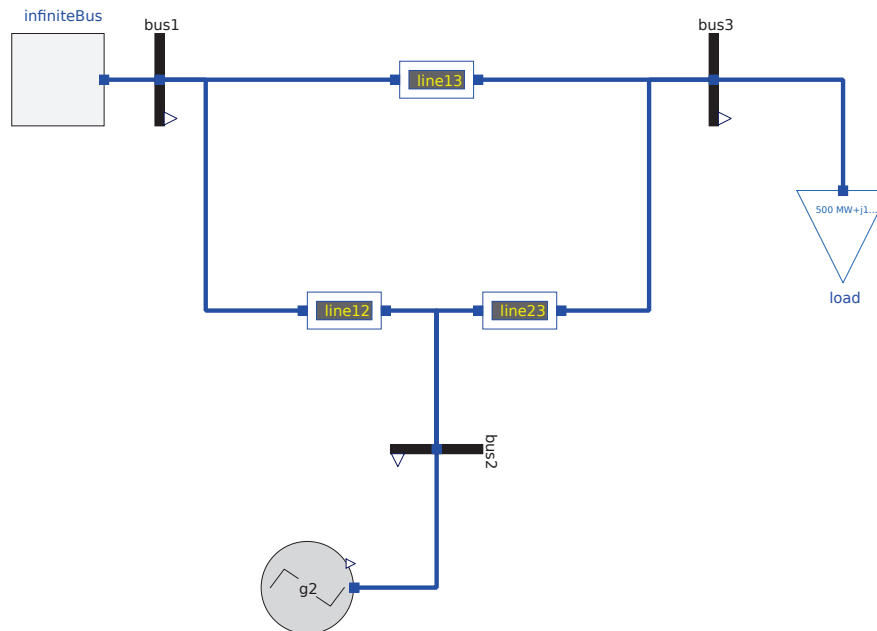


Figure 5.6: Three Buses monoblock model chosen to test the interface component under OMEdit

Figure 5.7 is an example of one of these tests. For this example, the connection between bus 2 and the generator is deleted. An interface component that has an input current and an output potential is connected via its pin to Bus2. Similarly, an interface component which has an output potential and an input current is connected to the generator. Then the inputs of one interface component are connected to the outputs of the other and vice versa. The 5.7 figure also zooms in on how the interface component is constructed, with a pin or plug to connect with

other components, and the inputs and outputs to receive and send information, as well as the derivatives as output/input to reconstruct the signal.

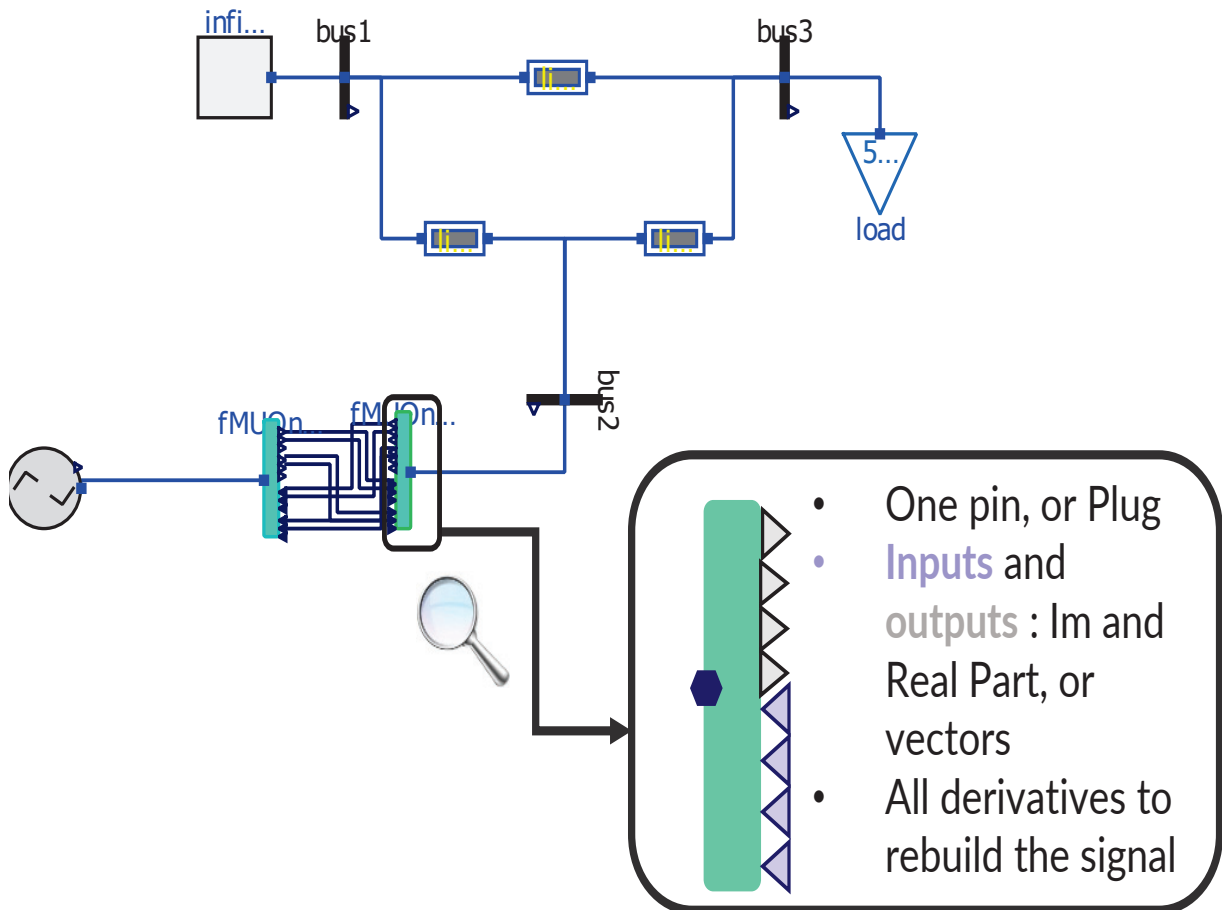


Figure 5.7: Three Buses model with interface components under OMEdit

Only the TS components passed this test, indeed the same results are obtained with the one-piece TS model (Figure 5.6) and that with the interfaces component (Figure 5.7). The results obtained with the interface components on the EMT model are not the same as those monoblock EMT. The components of the EMT interface as is are therefore not correct.

After investigation, one can realize that the artificial reconstruction of the signal proposed in the modelica library is not precise enough for EMT type models. This need for more precise reconstruction of the signal is due to the fact that the construction of OpenModelica only allows cutting before discretization. So when in the chapter 2 we could also pass what constituted the information at the

previous moment, here it is not possible to do it directly, we therefore choose to do it in a roundabout way by using a Taylor development. The changes in the interface components that have been made to overcome this problem are as follows:

- The signal is reconstructed more precisely, indeed a reconstruction of the signal for the inputs using a Taylor development of the first or second order is proposed.
- This reconstruction performed inside the interface component must be informed of the macro time step size. This macro time step will be chosen outside of the FMU by the user, it will be the meeting point between two sub-models. For more flexibility, we do not put a "parameter" character on this input, this leaves the possibility of using this model using variable macro time steps. A causal input Real called ΔT , corresponding to the macro time step has therefore been added to the component.

Remark 22. *As this reconstruction of the signal using the Taylor method is only used for the EMT interface component, the presence of the variable ΔT in the modelDescription.XML file is sufficient to indicate whether the model used is of the EMT type or TS type.*

The test in figure 5.7 is performed again with the new interface components. To test the Taylor reconstruction, the OpenModelica solver is forced to use chosen fixed time steps which are given as input as the macro time step. The results obtained with the monoblock EMT model (Figure 5.6) and that with the interface components (Figure 5.7) are the same. The EMT interface components are at this stage validated as not modifying the physics.

5.3.1.2 Causality of interface data

The sub-models thus obtained with validated interface components are exported as FMU. The zipped FMU folder is uncompressed to examine the modelDescription file. Indeed the modelDescription.XML needs to be consistent with the submodel, that is to say if the variables are all there, and if the causalities are well respected. Indeed, there are several possible causalities attributed to the variables of the model (local, parameter, input, output). Some FMI functions can only be used with variables having certain causality. For example the *fmi2SetReal* function can only be called (outside the initialization) with variables having an input type causality as arguments. The fact that causalities are correctly attributed is therefore a key element for the proper functioning of co-simulation.

It was noticed that the inputs and outputs of our submodels did not have the causality expected in the XML. Indeed, it turns out that an input or an output

is an input or an output of a model only if it is on the last layer of the model (i.e. not in a component). The inputs and outputs of our interface components are therefore not sufficient

Consequently, it is necessary to add external inputs and outputs to the sub-model. These new inputs and outputs are of the connector type and will be connected to the inputs and outputs of the interface components (the inputs and outputs of the interface components are also of the connector type). It is therefore necessary to add an input (resp. output) outside of any component for each input (resp. output) of the interfaces component. Then connect each input (resp. output) outside of any component to the corresponding input (resp. output) in the interface component.

Figure 5.8 shows on the left, a part of the modelDescription.xml of the FMU exported from the sub-model on the right. It can be seen that inputs and outputs have the expected causality.

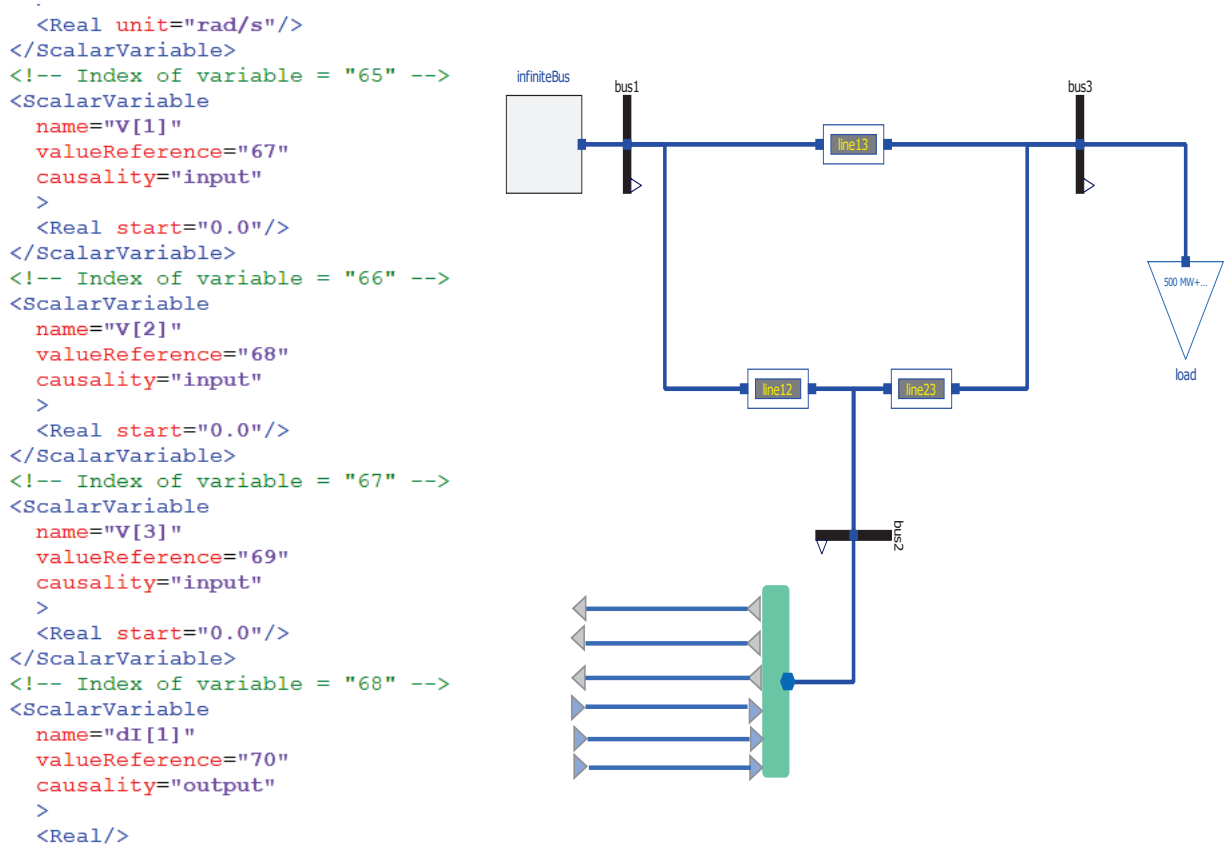


Figure 5.8: Test on the causality of values

Finally, the architecture of a model to be exported in FMU will be the one described in the class diagram 5.9. As can be seen, each subdomain is complemented by a number of interface components (m here) depending on the topology. Each interface component has a certain number k of inputs and n of outputs. Each input and output must be linked to an input (respectively to an output) which is on the last layer of the model. These inputs and outputs are connector type classes, so the model here has $m.n$ input on the last layer and $m.k$ outputs on the last layer. This architecture allows both the physics of the circuit to be correct and the causality to be correct in the ModelDescription.XML of the model.

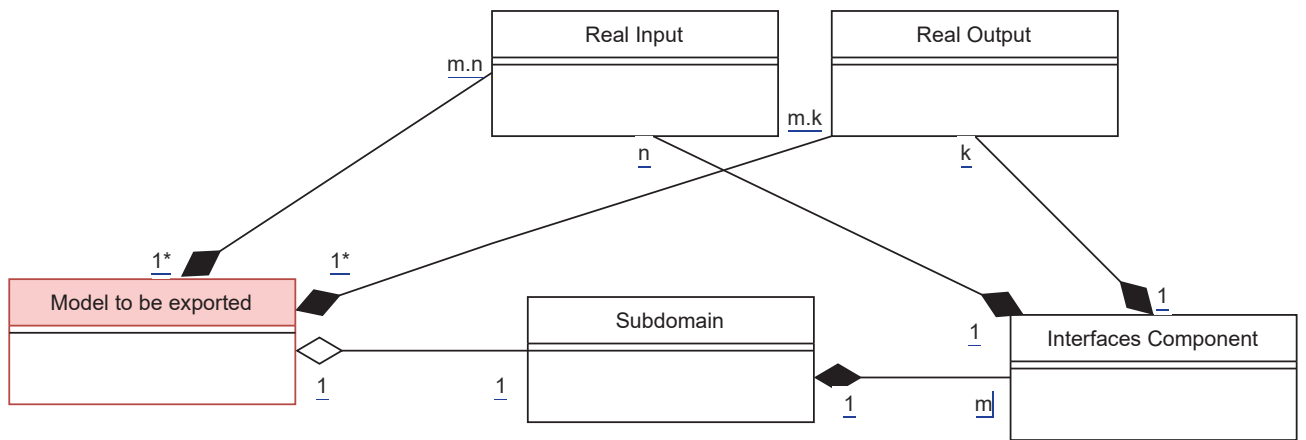


Figure 5.9: Class Diagram of a model to be exported

5.3.1.3 Other problems encountered

Despite our improvements to get usable FMUs exported from OpenModelica, we encountered many other problems related to the FMU export from OpenModelica, or the way the models were made. Here are some examples:

- As we wanted to use OpenModelica as much as possible, we wanted to use the Co-simulation type export. Indeed this would have allowed us to use the solvers proposed by OpenModelica. Unfortunately, the possibility of saving the state of an FMU proposed in the FMI standard in the form of the function *fmi2GetState* and the possibility of restarting the FMU from the state it was in at the start of the time step with the function *fmi2SetState* are only available (in its experimental version) for FMUs exported from OpenModelica since March 2022 for the unofficial version of OpenModelica and in June 2022 for the official version of OpenModelica. This ability to

store the state of the FMU is essential for performing a rollback, and is therefore essential for using the Schwarz method. We could therefore not use Co-simulation type FMUs for our Co-simulation until very recently.

- For export in Co-simulation the only solver available in non-experimental version is explicit Euler (CVODE is available in experimental version). Which might be a little skimpy for an EMT-like simulation.
- In the modelDescription.XML of an FMU, a *valueReference* is assigned to each variable. This *valueReference* is a key, only equal variables in the model can have the same *valueReference*. These keys are what are given as arguments to the *fmi2Get***** and *fmi2Set***** functions to indicate that these are the values to retrieve or which value to assign a value to. It is therefore essential that these *valueReference* are correctly assigned.

```

<ScalarVariable
  name="fMUOnePinPhaseurVinIout.iRe"
  valueReference="1"
  description="Current flowing from pin p to pin n"
  >
  <Real unit="A"/>
</ScalarVariable>
<Real/>
</ScalarVariable>
<!-- Index of variable = "2" -->
<ScalarVariable
  name="fMUOnePinPhaseurVinIout.yRe"
  valueReference="1"
  description="Output signal"
  >
  <Real/>
</ScalarVariable>
<ScalarVariable
  name="g2.g1.p.ir"
  valueReference="1"
  description="Real part of the current"
  >
  <Real/>
</ScalarVariable>
<ScalarVariable
  name="g2.pwPin.ir"
  valueReference="1"
  description="Real part of the current"
  >
  <Real/>
</ScalarVariable>
<ScalarVariable
  name="g2.ieesgo.imLeadLag.TF.na"
  valueReference="1"
  description="Size of Denominator of transfer function."
  variability="fixed"
  causality="calculatedParameter"
  >
  <Integer/>
</ScalarVariable>
<ScalarVariable
  name="fMUOnePinPhaseurVinIout.Name_fRe"
  valueReference="1"
  description="Name of flow variable"
  variability="fixed"
  causality="calculatedParameter"
  >
  <String/>
</ScalarVariable>

```

Figure 5.10: Modeldescription variable with the same reference value

Figure 5.10 shows a selected part of the FMU file `modeldescription.xml` exported from the model submodel 5.7 containing the generator. This FMU is exported from version 1.17 of OpenModelica installed under linux Ubuntu 20.041 LTS. As we can see, all the variables present in the figure carry the value Reference 1, except that the first four variables of the model are indeed equal, but not the last two. This defect renders this FMU unusable. To try to recover a usable FMU, we will try to export the model in FMU from another version of OpenModelica.

- To overcome the previous problem, we tried to export the same model to FMU from another version of OpenModelica. The Linux virtual machine with OpenModelica version 1.20.0 was chosen. This machine can be found ready to use on the OpenModelica website.

The `modelDescription.XML` exported from this OpenModelica version is correct: the *valueReference* are well assigned. The previous problem has therefore been corrected in this version of OpenModelica, however `.JSON` files are missing in the resource folder of the FMUs thus exported. These FMUs cannot therefore be instantiated with the *fmi2instantiate* function. This also renders FMUs exported from this version of OpenModelica unusable.

- We had various other problems that made this stage of the work last for several months. Many OpenModelica errors return an "index reduction failed" error even when the error has nothing to do with it. Problems of transition from acausal to causal which have been solved by modifying the basic models. Problems of choice of input, indeed certain variables cannot be chosen as inputs because there are hidden constraints in the models, and having an input bound to this constraint may put too many constraints on a value. This list of problems encountered is not exhaustive.

Remark 23. *At the time of writing this thesis, FMI 3 has just been made available. However, all the work done in this thesis has been done with FMI 2. Most tools allowing FMU export/import can currently only support FMI 1 and FMI 2 versions.*

5.3.1.4 Conclusions about FMI

FMI is a very practical tool. There are many tools that offer to export their models in FMU. Nevertheless, although the FMI interface is free, many paying tools do not allow the use of FMUs generated from their models elsewhere than in their tool (except against additional payment).

FMI is a tool that is constantly improving, Moreover, FMI3 was recently released with new possibilities such as easier export of virtual electronic control units, and

improved management of discrete events. The representation of the models thanks to the FMI standard is very clear and easily understandable, and the handling of this tool is quite instinctive.

In order to export a sub-model in FMU, for a Co-simulation, it is necessary to rework the sub-model, to add components specially designed for this purpose. This work was carried out under OpenModelica, a library meeting these needs for the TS models and for the EMT models was created.

We are less enthusiastic with the OpenModelica FMU export.

Indeed, the OpenModelica FMU export seems quite experimental. It seems that there is still work to be done before the use of FMUs exported from OpenModelica can be done serenely.

5.4 Orchestrator

In the previous section, it was explained how to create from OpenModelica units to be simulated: the FMUs. To use these standardized models, there are two main possibilities:

- Re-import the FMUs into a tool that allows it: several tools offer to re-import FMUs possibly generated from other tools, such as Dymola or OpenModelica (experimentally), the FMUs are then transformed back into a modelica model and can be simulated as a standard model. A multi-FMU structure and its parametrization can also be re-imported into many tools, using the System Structure and Parameterization (SSP) standard developed by the modelica association [13]. This way of doing things does not allow us to implement our method for the reasons mentioned in 5.2.
- Use the functions proposed by FMI and exported in an FMU (via .dll) to use the FMUs as a black box and orchestrate the simulation: This way of using the FMUs is used for co-simulation on an industrial scale, in particular by DACCOSIM [35], a co-simulation tool developed by EDF and central-supelec. This tool, which is very efficient in terms of task placement, uses a Newton-Raphson method for co-initialization and during co-simulation, if the error is too large, the tool performs a roll back by reducing the size of its macro time steps. This tool demonstrates that it is possible to do large-scale co-simulation using FMI. However, this very powerful tool with software FMUs like Dymola will encounter the problem of not being able to perform roll back with a CS FMU that would be exported from OpenModelica. Indeed, Daccosim is created to co-simulate CS-type FMUs, moreover the time step reduction method aims to make the error operator contractant (cf chapter 4).

We have chosen to apply another method which gets rid of the contractance constraint of the error operator, moreover we absolutely wish to use the free tools of OpenModelica.

We are going to use the FMUs in a second step, for that we are going to create an orchestrator containing the methods presented previously. We are therefore going to develop a simulation platform whose heart will be our numerical method. For this we center its implementation on a Master/Slave architecture where the Master will be an orchestrator in charge of asking the slaves to simulate their system and in charge of providing the interfaces to each one. We will first talk about the general structure of the platform and explain the communication process. Secondly, we will present the specificities related to FMUs.

5.4.1 General Structure

In view of all the difficulties encountered when exporting FMU with OpenModelica, we wanted to create an Orchestrator that is as general and flexible as possible. That is, it must be able to co-simulate CS FMU, ME FMU and DAE models written in C++ in case the user does not have access to the correct FMUs.

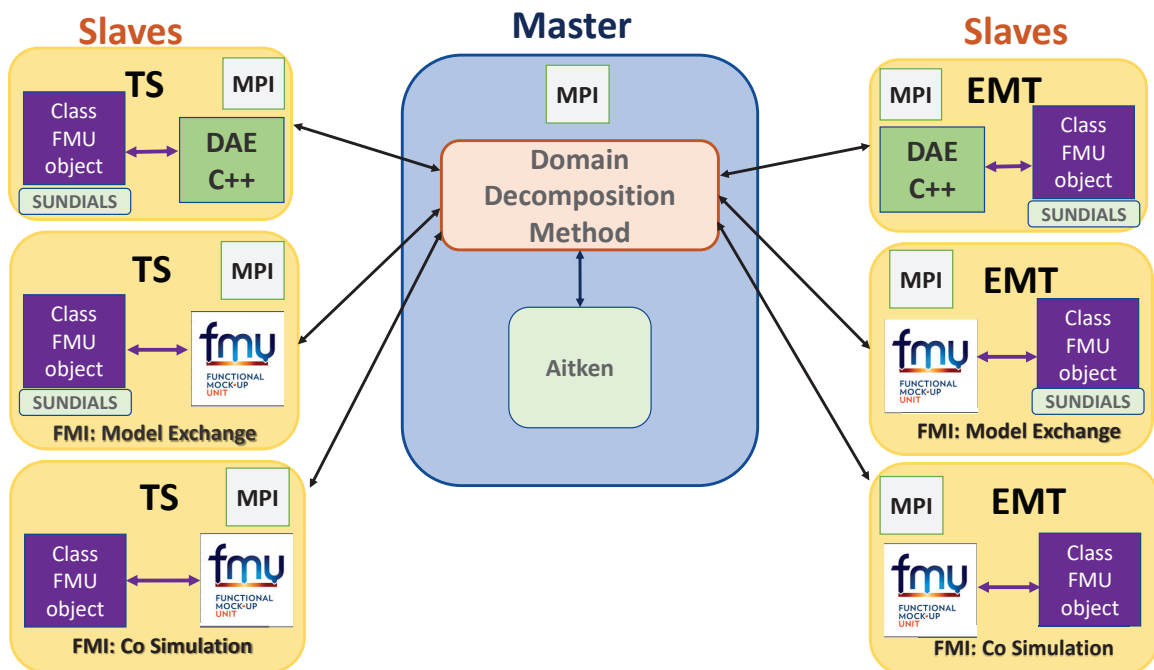


Figure 5.11: Co-simulation platform architecture with MPI Master-Slaves communications and using local solver in the FMI standard or C++ DAE functions.

Figure 5.11 present this platform: The platform is designed in a Master-Slave approach. The Master is in charge of the co-simulation algorithm: the RAS method, as well as the Aitken convergence acceleration method. The Master is also responsible for orchestrating the simulation. Indeed it must call the EMT and TS slave models, give orders concerning the time steps to play: the start time and the duration of this macro time step. The Master also has the role of recovering the information calculated in the various slave models, as well as transforming this information if necessary (EMT-TS translation, see chapter 3), then the master is responsible for redistributing these information. Finally, the master is the one who indicates that the simulation is over.

There is a homogenization of the external representation of the slave models. Indeed, each sub-domain is represented by a slave in the form of a black box, the interest being to have a similar external representation for all the sub-domains even if they are not coded in the same way. Local resolutions are performed on the slave side via an instance of the FMUobject architecture. Each slave contains an instance of a C++ FMUobject class that embeds a local DAE solver such as IDA of Sundials or the embedded FMU solver if the C++ FMU object contains a CS FMU. The slave is also responsible for instantiating the FMU when it applies.

To summarize the general operation of the platform represented by the figure 5.11: each slave instance only receives commands from the master, reacts internally to these commands by carrying out the tasks that this implies according to its internal operating coding, and the only information they can send back to the master is the interface values they have calculated internally or their status.

These command exchanges are issued using point-to-point MPI communication routines. Communication is described below.

5.4.1.1 Communication protocol for the platform

Communication between the slaves and the master is initiated as an MPI (Message Passing Interface) process. MPI is a standard for efficiently communicating between multiple processes, it contains a library of functions and is really efficient for performing parallel simulations. The master and slaves communicate using a point-to-point communication routine.

In this architecture, one MPI process is used per calculation unit. Each subdomain (slave) is a computing unit and the master is one too. The use of MPI makes it possible to simplify the management of exchanges, in fact each process has an identifier. The master has the identifier 0.

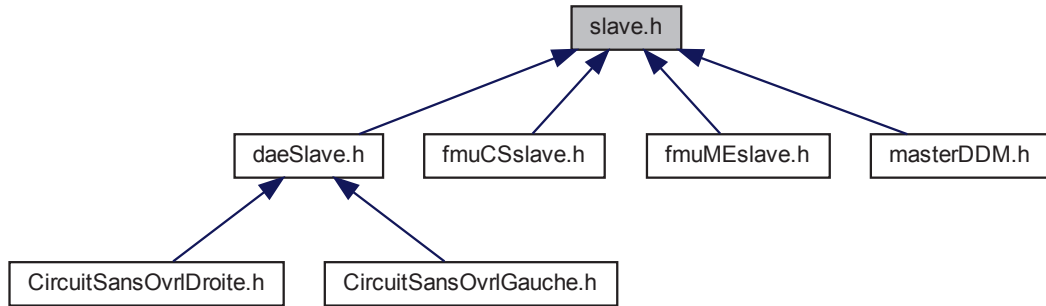


Figure 5.12: Class Diagram of the Architecture

The different classes are created in C++. We can thus use the object side of this language (Figure 5.12). Indeed, an abstract parent class named “slave” is created and all the slave instances as well as the master instance inherit from this class. This type of architecture not only makes it possible to reuse a type of class rather than recreating each time a basic structure, but also to use polymorphism. Indeed our final structure will thus be able to have a certain number of class derived from the "slave" class without having to specify what type of class it is exactly and by designating them only by its MPI process identifier.

Figure 5.13 gives more details on this architecture. As can be seen, all classes derive from a "slave" class having certain attributes and functions needed by all classes. Then each class has new attributes depending on its type. The Master will therefore have the Schwarz and Aitken methods as well as the "sendCommandFmu()" function which will allow him to orchestrate the simulation. The different slaves will have methods allowing them to receive these instructions given by the orchestrator as well as functions allowing internal resolution such as "doACosimulationStep()" which is a method specific to each type of slave.

The final subdomains represented here by the “CircuitSansOvrlDroite” and “CircuitSansOvrlGauche” classes are final slave classes. Here, these classes inherit from the "daeSlave" class. Each of these classes has, in addition to all the attributes and functions of the "daeSlave" class, a system of algebraic differential equations written in C++ which is specific to it and which represents the part of the circuit which it is supposed to simulate. In this example, an instance of the "masterDDM" class will have an MPI process with identifier 0, an instance of the "CircuitSansOvrlDroite" class will have an MPI process with identifier 1 and

an instance of the "CircuitSansOvrlGauche" class will have a MPI process with identifier 2.

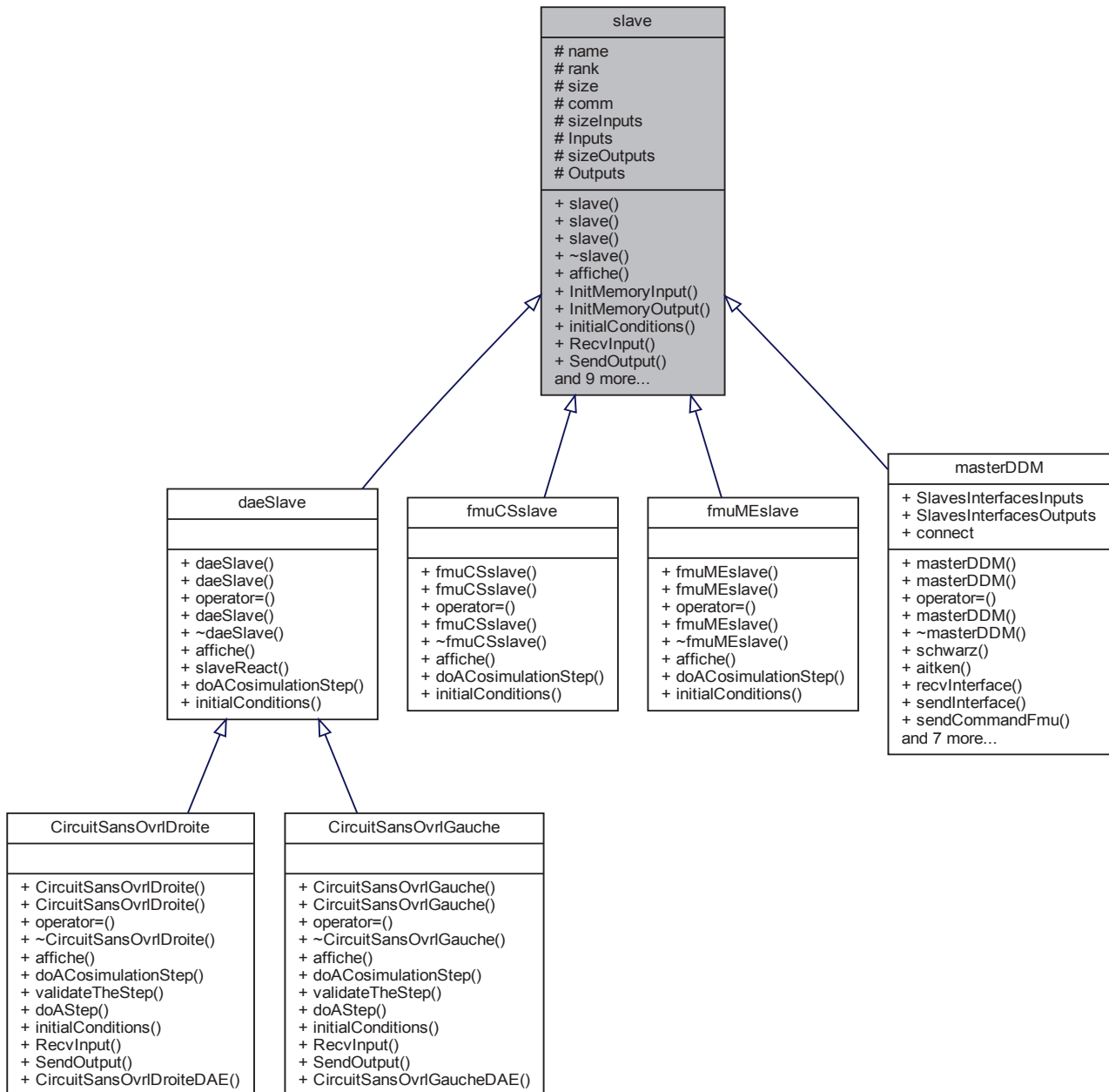


Figure 5.13: Detailed Class Diagram of the Architecture.

5.4.2 FMU type slave

To use FMU functions as well as the types (`fmi2Real`, `fmi2char...`) defined by the FMU standard, one must also retrieve 3 Headers available on the site dedicated to the FMI standard: `fmi2FunctionTypes.h`, `fmi2Functions.h`, `fmi2TypesPlatform.h`. The `fmuCSslave` and `fmuMEslave` classes are based on information retrievable in the `modelDescription.XML` (retrieved during export see previous section 5.3). They also contain an FMU component (type **`fmi2Component`**) that functions as a black box within the slave class itself as well as functions that are used to interact with this FMU component. To instantiate an FMU component `fmu2Component`, you must call the *`fmi2instanciate`* function which requires information extracted from `modelDescription.XML` as an argument. The first step therefore consists in recovering the functions of the DLL libraries (exported in the `.FMU` zip folder) . It is then necessary to retrieve the various information via the `modelDescription.XML`. To do this, the `libXML` library was used. We seek causality to find the inputs and outputs that we store in a map using the name of the variables as a key. Indeed, we have chosen to automatically link the inputs and outputs rather than letting the user do it. The reference value assigned to a value being specific to an FMU export, the only indication that can be used to identify the variables is their name.

At the end of this process we obtain a class instance having:

- a map containing the `ValueReference` of each input and their names: the name is useful to identify the inputs necessary for this sub-model and to inform the master during the initialization of the co-simulation. The `ValueReference` serve as arguments for the *`fmi2SetReal`* function. The variable's name is a general key and its `ValueReference` is a local key.
- a map containing the `ValueReference` of each output and their names. The name serves as a global key during initialization to inform the master of the calculated outputs in this slave class. The name serves as the local key and is supplied as an argument to the *`fmi2GetReal`* function.
- a map containing the `ValueReference` of all the model values and their names in order to retrieve all the results, useful to call the *`fmi2GetReal`* function at the end of a time step (after convergence) to get the results. The name lets the user know what the value stands for.
- an instance of an **`fmi2Component`**. This black box is the heart of the class, it contains the DAE of the model (not accessible).
- the functions which are necessary to communicate with the **`fmi2Component`**, there are functions common to FMU CS and FMU ME, and many specific

functions which can only be used with one type of FMU. Figure 5.14 is a table taken from the FMI 2 specification, it gives a list of common available functions for FMU CS and FMU ME.

Function	start, end	instantiated	Initialization Mode	Event Mode	Continuous-Time Mode	terminated	error	fatal
fmi2GetTypePlatform	x	x	x	x	x	x	x	
fmi2GetVersion	x	x	x	x	x	x	x	
fmi2SetDebugLogging		x	x	x	x	x	x	
fmi2Instantiate	x							
fmi2FreeInstance		x	x	x	x	x	x	
fmi2SetupExperiment		x						
fmi2EnterInitializationMode		x						
fmi2ExitInitializationMode			x					
fmi2Terminate				x	x			
fmi2Reset		x	x	x	x	x	x	
fmi2GetReal			2	x	x	x	7	
fmi2GetInteger			2	x	x	x	7	
fmi2GetBoolean			2	x	x	x	7	
fmi2GetString			2	x	x	x	7	
fmi2SetReal		1	3	4	5			
fmi2SetInteger		1	3	4				
fmi2SetBoolean		1	3	4				
fmi2SetString		1	3	4				
fmi2GetFMUstate		x	x	x	x	x	7	
fmi2SetFMUstate		x	x	x	x	x	x	
fmi2FreeFMUstate		x	x	x	x	x	x	
fmi2SerializedFMUstateSize		x	x	x	x	x	x	
fmi2SerializeFMUstate		x	x	x	x	x	x	
fmi2DeSerializeFMUstate		x	x	x	x	x	x	
fmi2GetDirectionalDerivative			x	x	x	x	7	

Figure 5.14: FMI 2 functions

FMU Co Simulation

Function	FMI 2.0 for Co-Simulation									
	start, end	instantiated	Initialization Mode	stepComplete	stepInProgress	stepFailed	stepCanceled	terminated	error	fatal
fmi2SetRealInputDerivatives		x	x	x						
fmi2GetRealOutputDerivatives				x		8	x	x	7	
fmi2DoStep				x						
fmi2CancelStep					x					
fmi2GetStatus				x	x	x		x		
fmi2GetRealStatus				x	x	x		x		
fmi2GetIntegerStatus				x	x	x		x		
fmi2GetBooleanStatus				x	x	x		x		
fmi2GetStringStatus				x	x	x		x		

Figure 5.15: FMI 2 functions for Cosimulation

The **fmi2Component** can call its own internal solver. So the main important function proper to an FMU CS is *fmi2DoStep*. Figure 5.15 is a table available in the FMI 2 specifications, it gives the functions specific to the FMU CS.

FMU Model Exchange

Function	FMI 2.0 for Model Exchange									
	start, end	instantiated	Initialization Mode	Event Mode	Continuous-Time Mode	terminated	error	fatal		
fmi2EnterEventMode				x	x					
fmi2NewDiscreteStates				x						
fmi2EnterContinuousTimeMode				x						
fmi2CompletedIntegratorStep					x					
fmi2SetTime				8	x					
fmi2SetContinuousStates					x					
fmi2GetEventIndicators			x	x	x	x	7			
fmi2GetContinuousStates			x	x	x	x	7			
fmi2GetDerivatives			x	x	x	x	7			
fmi2GetNominalsOfContinuousStates		x		x	x	x	7			

Figure 5.16: FMI 2 functions for Model Exchange

The **fmi2Component** can not call an internal solver. To simulate the system, one must retrieve an estimate of the states and derivatives using the *fmi2GetDerivatives* function and simulate the system using a solution external to the **fmi2Component**. Figure 5.16 is a table available in the FMI 2 specifications, it gives the functions specific to the FMU ME.

5.4.2.1 FMU slave Management

Figure 5.17 is a pseudo code representing the co-simulation between two FMU slave types, the left slave is an FMU ModelExchange and the right one is an FMU CoSimulation slave. The reaction of the FMUs to the master command *doACosimulationStep* is not the same. Indeed when the FMUCSSlave needs to reset its FMU component and can call its intern solver, the FMUMESlave estimates a derivative and the simulation is performed with an chosen external solver.

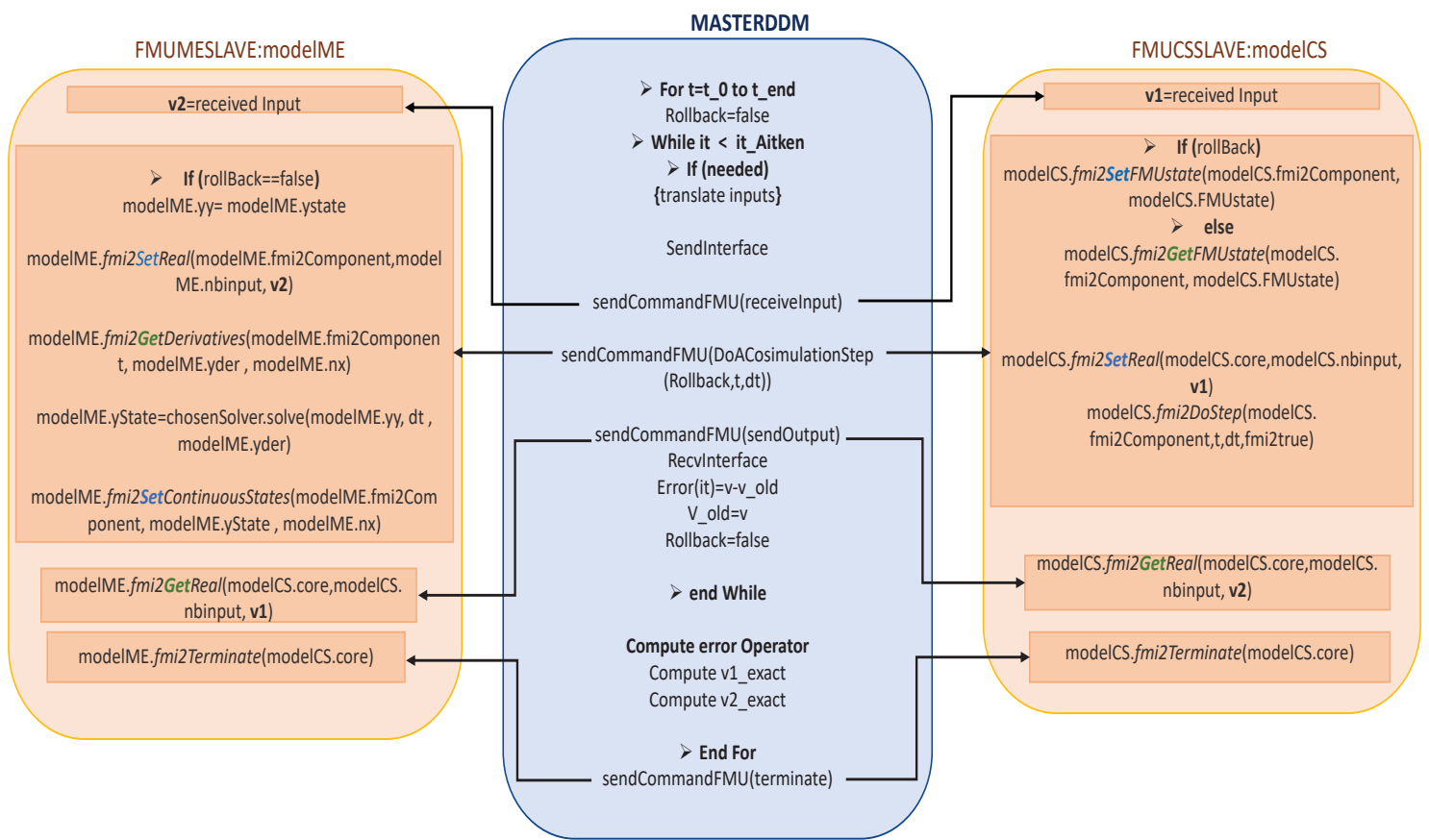


Figure 5.17: Pseudo Code representing the Co simulation of two FMU slave

5.4.2.2 C++ wrapping test

In the platform test steps, one of the first is to test the C++ wrapper, that is to say that we recover the information from the `modelDescription.XML`, that we recover the functions of the DLLs and that the call to these functions is correct.

For this, a known ODE system as small as possible is chosen. Indeed, it helps focus on the wrapper and minimizes the possibility that an error is not related to the C++ wrapper.

The chosen system is the Lokta Volterra, a system with two equations and two unknowns which models the evolution of populations of prey and predators. This model was also used to test the MasterSim platform[88]. By noting x the population of preys and y the population of predators, $\alpha, \beta, \delta, \gamma$ are constants representing the mortality, the reproduction of the preys and predators. This system is defined as:

$$\begin{cases} \dot{x}(t) &= x(t)(\alpha - \beta y(t)) \\ \dot{y}(t) &= y(t)(\delta x(t) - \gamma) \\ x(t_0) &= x_0 \\ y(t_0) &= y_0 \end{cases} \quad (5.1)$$

We place the first equation in a model with x calculated in this model and y as input. The second model consists of the second equation and has x as input. We thus obtain two FMUs which are tested in Co-simulation and in Model-Exchange.

Figure 5.18 shows the results of the cosimulation of the two FMUs CS (the one in the middle) and the two FMUs ME (the one at the bottom) using the platform. these results are compared to the simulation performed with the entire model in OMEdit (top). The C++ wrapper works correctly.

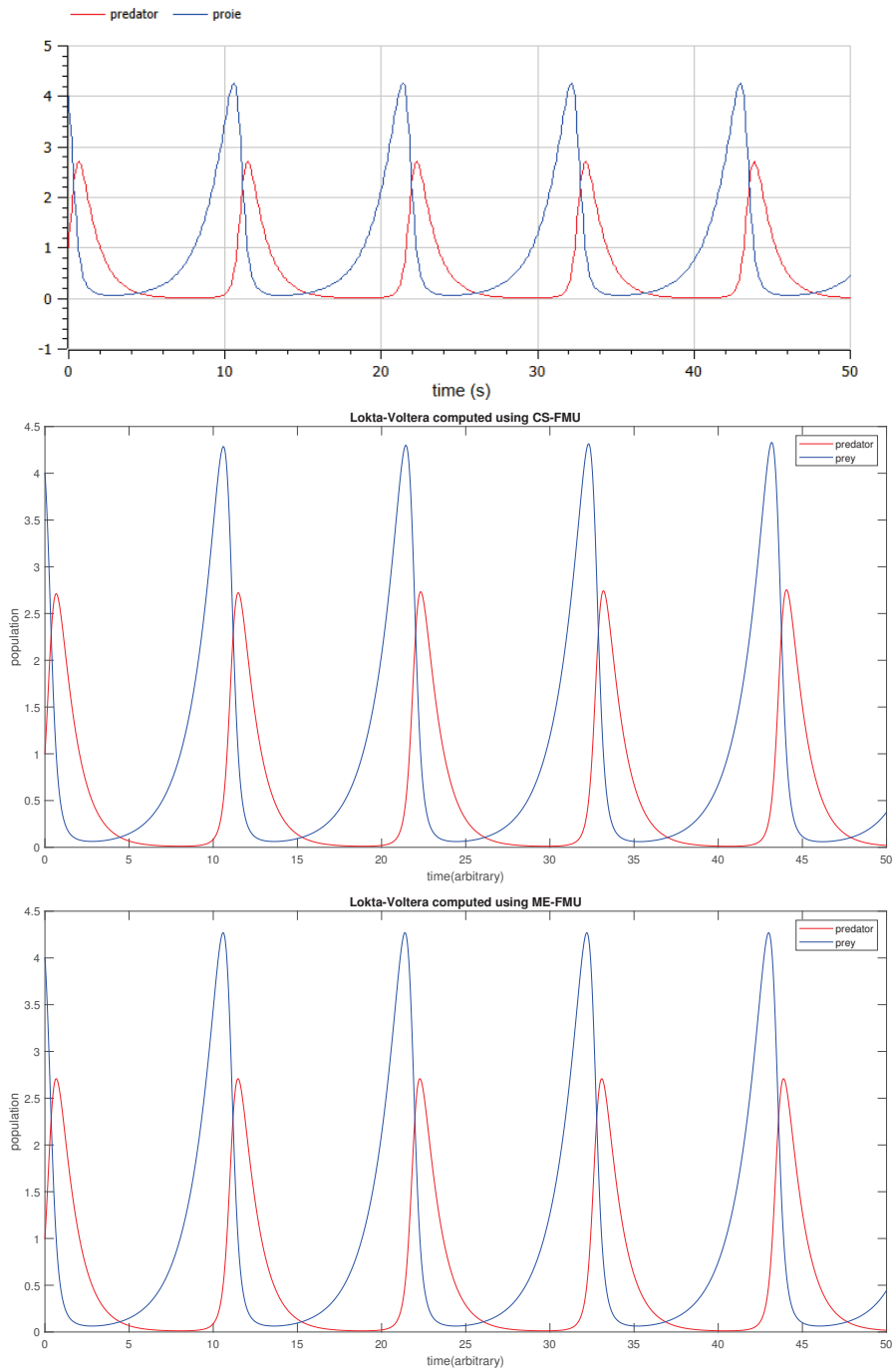


Figure 5.18: C++ wrapper validation results using the Lokta-Voltera with $\alpha = \frac{2}{3}$, $\beta = \frac{4}{3}$, $\delta = 1$, $\gamma = 1$

5.4.3 Conclusion About the Orchestrator

The platform under development is based on a master/slave architecture, it was designed to be adapted to the use of FMU. Indeed, the fundamental concept of the method is to post-process the sequence of interface solutions generated by the domain decomposition solver. It can use different boundary conditions for acceleration as long as they are linear in the variables (i.e. Dirichlet, Neumann, Robin, [102],...). This post-processing is particularly interesting here because the process can be used in a non-intrusive way, which is particularly suitable when the models used are, like the FMUs, in the form of a black box. To exchange information, the point-to-point communication routine of the MPI is used. The designed architecture is very general because it can use FMUs Model Exchange, FMUs Co Simulation or DAE written in C++ as slaves. However, all these slaves are perceived by the master as having the same form and are treated the same way, so the differentiation is only made within the slave instance.

Chapter 6

Conclusions and perspectives

With the integration into the Power Grid of power electronic type technologies, faster dynamics than before are appearing in the network. This evolution of the network must be accompanied by an evolution of the methods used to simulate it. In this perspective, EMT-TS co-simulation is a promising method that is regularly explored. Indeed, as the different technologies present in the network involve several types of transients, simulating it with several types of simulation seems logical. In this co-simulation method, limited parts of the network involving fast transients are simulated with EMT-like simulation and the rest of the network, is simulated with TS-like simulation. Thus, the goal of the maneuver is to take advantage of both the level of detail offered by the EMT-type simulation and both the calculation speed of the TS-type simulation.

In this PhD, the practicality of domain decomposition methods has been put at the service of this subject of co-simulation. Firstly, the Restrictive Additive Schwarz type method was applied to electrical circuits. The convergence of these methods has been studied for linear RLC circuits or if there are nonlinear components, for linearized systems around each time step. In the homogeneous case, convergence or divergence is purely linear, which allowed us to apply Aitken's convergence acceleration technique. Thanks to this acceleration, the results of the simulation are obtained in a limited number of iterations, whether the method converges or diverges. Since, thanks to Aitken's convergence acceleration technique, we obtain the converged solution even in the case of a divergent method, taking into account the splitting constraint to have a convergent method is no longer relevant.

In a second step, the method was modified to be able to be used in the heterogeneous case. The protocol was adapted to the time step difference between the EMT simulation and the TS simulation. Translation operators were chosen and improved to meet our requirements: these translations had to be as precise as possible, these translations should not be too expensive, translations that do not add additional constraints on the time steps used and therefore on the protocol and

finally linear translations in order to be able to keep the convergence/divergence linear and therefore the advantage of being able to use the Aitken's acceleration of the convergence technique. The strategy of using the Aitken technique to accelerate convergence was designed to best adapt to the protocol induced by the heterogeneity of the co-simulation. Hence, a heterogenous EMT-TS RAS method accelerated by Aitken's convergence acceleration technique was obtained.

Afterwards, It was shown that the Restrictive Additive Schwarz method is a special case of a general method called Dynamic Iteration. We then demonstrate that dynamic iteration with the RAS splitting method can be accelerated using the Aitken's convergence acceleration technique and in particular multiple time steps can be accelerated at the same time. This other protocol is then compared to the protocol used in this PhD so far and we see that it is particularly interesting when a nonlinear component is in the circuit or when programmed variable time steps are used. Indeed this protocol makes it possible to obtain in these cases the converged solution without needing to make more iterations in order to manage the nonlinear components or variable time steps.

Throughout this thesis, we have explored OpenModelica tools, a suite of tools that has potential and whose future developments should overcome the difficulties we encountered.

Finally, a platform based on the previously developed methods has been designated and is being developed at SuperGrid Institute. This platform is intended to be as general as possible, using either models in the form of functional mock-up unit (FMU) co-simulation (using the export tool's solver) or model exchange (resolution with an external solver such as those offered by SUNDIALS) or DAE coded in C++. This platform is designed in a master/slave approach and uses the point-to-point communication routine offered by mpi.

Perspectives

It would be interesting to study large overlap.

Indeed, this would make it possible to deepen the study of the impact of the overlap on the convergence of the method but also to compare the two different representations on the overlap and thus to estimate the information which was lost during the passage of the data. .

In this thesis the quantities exchanged were either potentials or currents, and those measures were exchanged using Dirichlet type conditions. These conditions could be improved, one idea would be to impose that a quantity be invariant, for example the power.

The study of a significant overlap could also give indications on the relevance of the partitioning, for example if the solution given by the EMT simulation is the

same at a point of the network as that given by the TS simulation, one could conclude that the use of EMT simulation in this location is not essential and the EMT sub domain can be reduced.

The partitioning could also be reworked from different points of view. In particular, the methods presented in the thesis make it possible to overcome the constraint that the error operator of the method is contractant (i.e. that the method converges), which leaves greater freedom in the choice of partitioning.

In this way, a point of view could be a partitioning based on a partitioning study of heterogeneous graphs, in order to minimize the computational costs as much as possible.

One could also use a dynamic partitioning which would make it possible to enlarge the EMT part when an event would have been detected or on the contrary to reduce this part when no fast dynamic is detected. The constraint of this idea being that it is necessary to have large models in TS version and in EMT version and not to work with fixed models. Indeed, by using for example FMUs, it would certainly be necessary to export too many of them and in addition to instantiate two new models with each evolution of the partitioning, which seems quite irrational.

Bibliography

- [1] S. Abhyankar and A. J. Flueck. An Implicitly-Coupled Solution Approach for Combined Electromechanical and Electromagnetic Transients Simulation. In *2012 IEEE POWER AND ENERGY SOCIETY GENERAL MEETING*, 2012. 1.5.2
- [2] G. Ali, A. Bartel, M. Brunk, and S. Schoeps. A Convergent Iteration Scheme for Semiconductor/Circuit Coupled Problems. In Michielsen, B. and Poirier, J.R., editor, *Scientific Computing in Electrical Engineering (SCEE 2010)*, volume 16 of *Mathematics in Industry-Cham*, pages 233–242, 2012. 4.1.2
- [3] U. D. Annakkage. *Transient Analysis of Power Systems: Solution Techniques, Tools and Applications*, chapter Dynamic System Equivalents. IEEE Press. Wiley, 2015. 1.5.1.2, 1.5.3.2
- [4] P. Aristidou. *Time-domain simulation of large electric power systems using domain-decomposition and parallel processing methods*. PhD thesis, 06 2015. 1.4.1, 1.5.1.3
- [5] P. Aristidou, D. Fabozzi, and T. Van Cutsem. A Schur Complement Method for DAE Systems in Power System Dynamic Simulations. In J. & al Erhel, editor, *Domain Decomposition Methods in Science and Enigineering XXI*, volume 98 of *Lecture Notes in Computational Science and Engineering*, pages 719–727, 2014. 1.4.1
- [6] P. Aristidou, D. Fabozzi, and T. Van Cutsem. Dynamic Simulation of Large-Scale Power Systems Using a Parallel Schur-Complement-Based Decomposition Method. *IEEE Trans. Parallel Distrib. Syst.*, 25(10):2561–2570, OCT 2014. 1.4.1
- [7] P. Aristidou, S. Lebeau, and Th. Van Cutsem. Power System Dynamic Simulations Using a Parallel Two-Level Schur-Complement Decomposition. *IEEE Trans. Power Syst.*, 31(5):3984–3995, 2016. 1.4.1

- [8] P. Aristidou and T. Van Cutsem. Algorithmic and computational advances for fast power system dynamic simulations. IEEE Power and Energy Society General Meeting PESGM. IEEE, 2014. 1.4.1
- [9] P. Aristidou and T. Van Cutsem. Dynamic simulations of combined transmission and distribution systems using parallel processing techniques. In *2014 Power Systems Computation Conference (PSCC)*. Inst Elect & Elect Engineers, IEEE, 2014. 1.4.1
- [10] P. Aristidou and T. Van Cutsem. A parallel processing approach to dynamic simulations of combined transmission and distribution systems. *Int. J. Electr. Power Energy Syst.*, 72(SI):58–65, 2015. 1.4.1
- [11] M. Arnold. Constraint partitioning in dynamic iteration methods. *Z. Angew. Math. Mech.*, 81(3):S735–S738, 2001. 4.1.2
- [12] M. Arnold and M. Gunther. Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT*, 41(1):1–25, 2001. 4.1.2
- [13] Modelica association. System structure and parameterization, 2019. 5.4
- [14] Modelica association. Modelica language specification version 3.6-dev, 2022. 5.2.1
- [15] D. Athaide, J. Qin, and Y. Zou. MATLAB/Simulink-Based Electromagnetic Transient-Transient Stability Hybrid Simulation for Electric Power Systems with Converter Interfaced Generation. In *2019 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–6, 2019. 1.5.3.1
- [16] S. Bacha, Iulian Munteanu, and Antoneta Iuliana Bratcu. *Generalized Averaged Model*, pages 97–147. Springer London, London, 2014. 1.3
- [17] A. Bartel, M. Brunk, M. Guenther, and S. Schoeps. Dynamic Iteration for coupled problems of electrical circuits and Distributed Devices. *SIAM J. Sci. Comput.*, 35(2):B315–B335, 2013. 4.1.2
- [18] A. Bartel, M. Brunk, and S. Schoeps. On the convergence rate of dynamic iteration for coupled problems with multiple subsystems. *J. Comput. Appl. Math.*, 262:14–24, 2014. 4.1.2
- [19] A. Bartel and M. Guenther. PDAEs in Refined Electrical Network Modeling. *SIAM Review*, 60(1):56–91, 2018. 4.1.2

- [20] M. Baudette, M. Castro, T. Rabuzin, J. Lavenius, T. Bogodorova, and L. Vanfretti. Openipsl: Open-instance power system library — update 1.5 to “itesla power systems library (ipsl): A modelica library for phasor time-domain simulations, 2018. 5.2.1
- [21] T. Blochwitz, M. Otter, J. Åkesson, M. Arnold, Ch. Clauss, H. Elmqvist, M. Friedrich, A. Junghanns, J. Mauss, D. Neumerkel, H. Olsson, and A. Viel. Functional mockup interface 2.0: The standard for tool independent exchange of simulation models. In *Proceedings of the 9th International Modelica Conference*, pages 173–184. The Modelica Association, 2012. 1.4.1
- [22] M. A. Botchev, I. V. Oseledets, and E. E. Tyrtysnikov. Iterative across-time solution of linear differential equations: Krylov subspace versus waveform relaxation. *Comput. Math. Appl.*, 67(12):2088–2098, JUL 2014. 4.1.2
- [23] X. Cai and M. Sarkis. A restricted additive schwarz preconditioner for general sparse linear systems. *SIAM J. Sci. Comput.*, 21:792–797, 1999. 2.1
- [24] R. J.G.B. Campello, D. Moulavi, and J. Sander. Density-Based Clustering Based on Hierarchical Density Estimates", booktitle="Advances in Knowledge Discovery and Data Mining. pages 160–172. Springer Berlin Heidelberg, 2013. 1.5.1.2
- [25] I. Cortes Garcia, H. De Gersem, and S. Schöps. A structural analysis of field/circuit coupled problems based on a generalised circuit element. *Numerical Algorithms*, 83:373–394, 2020. 5.2.2
- [26] Q. Cossart. *Tools and Methods for the Analysis and Simulation of Large Transmission Systems Using 100 % Power Electronics*. PhD thesis, Ecole nationale supérieure d’arts et métiers - ENSAM, Sep 2019. 1.4.2, 1.5.1.2
- [27] J. Dadallage, S.and Dyck, N. Kroeker, A. Poersch, and L. Unruh. E-tran plus software. 1.4.2
- [28] T. Demiray. *Simulation of Power System Dynamics using Dynamic Phasor Models*. PhD thesis, SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH, 2008. 1.3
- [29] Fröderer der Energie-und Informationstechnik für zukunftsfähige Netze Aachen e.V. association. Modelica package for power system models, 2018. 5.2.1
- [30] HW Dommel. Digital computer solution of electromagnetic transients in single- and multiphase networks. *IEEE TRANSACTIONS ON POWER APPARATUS AND SYSTEMS*, PA88(4):388–&, 1969. 2.3.1

- [31] Y. Eguillon, B. Lacabanne, and D. Tromeur-Dervout. IFOSMONDI: A Generic Co-simulation Approach Combining Iterative Methods for Coupling Constraints and Polynomial Interpolation for Interfaces Smoothness. In SCITEPRESS Science and Technology Publications, editors, *Proceedings of the 9th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, pages 176–186, 2019. 1.5.2, 2.1
- [32] Y. Éguillon, B. Lacabanne, and D. Tromeur-Dervout. IFOSMONDI Co-simulation Algorithm with Jacobian-Free Methods in PETSc. *Engineering with computers*, 2022. 1.5.2, 2.1
- [33] B. Engquist and H-K. Zhao. Absorbing boundary conditions for domain decomposition. *Appl. Numer. Math.*, 27(4):341–365, 1998. 2.1, 2.2.1, 2.2.1
- [34] J. Evora Gomez, J. Cabrera, J-P. Tavella, S. Vialle, E. Kremers, and L. Frayssinet. Daccosim ng: co-simulation made simpler and faster. pages 785–794, 02 2019. 1.4.1
- [35] J. Évora Gómez, J. Juan Hernández Cabrera, J.-P. Tavella, S. Vialle, E. Kremers, and L. Frayssinet. Daccosim NG: co-simulation made simpler and faster. In *13th International Modelica Conference 2019*, 2019. 5.4
- [36] D. M. Falcao, E. Kaszkurewicz, and H. L. S. Almeida. Application of parallel processing techniques to the simulation of power system electromagnetic transients. *IEEE Transactions on Power Systems*, 8(1):90–96, Feb 1993. 1.5.1.1, 1.5.1.3
- [37] P. Fritzson and P. Bunus. Modelica—a general object-oriented language for continuous and discrete-event system modeling and simulation. In *Proceedings 35th Annual Simulation Symposium. SS 2002*, pages 365–380. IEEE, 2002. 5.2.1
- [38] P. Fritzson and V. Engelson. Modelica—a unified object-oriented language for system modeling and simulation. In *European Conference on Object-Oriented Programming*, pages 67–90. Springer, 1998. 5.2.1
- [39] M. J. Gander. Schwarz methods over the course of time. *Electronic Transactions on Numerical Analysis*, pages 228–255, 2008. 1
- [40] M. J. Gander, L. Halpern, and F. Magoules. An optimized Schwarz method with two-sided Robin transmission conditions for the Helmholtz equation. *Int. J. Numer. Meth. Fluids*, 55(2):163–175, SEP 20 2007. 2.2.1

- [41] M. J. Gander, P. M. Kumbhar, and A. E. Ruehli. Asymptotic Analysis for Overlap in Waveform Relaxation Methods for RC Type Circuits. *SIAM J. Sci. Comput.*, 84(1), 2020. 1.4.1, 1.5.1.2
- [42] M. J. Gander, F. Magoulès, and F. Nataf. Optimized Schwarz Methods without Overlap for the Helmholtz eq. *SIAM J. Sci. Comput.*, 24(1):38–60, 2002. 2.2.1
- [43] M.J. Gander, L. Halpern, and F. Nataf. Optimal Convergence for Overlapping and Non-Overlapping Schwarz Waveform Relaxation. In C-H. Lai, P. Bjørstad, M. Cross, and O. Widlund, editors, *Eleventh international Conference of Domain Decomposition Methods*. ddm.org, 1999. 2.2.1
- [44] M. Garbey and D. Tromeur-Dervout. A parallel adaptive coupling algorithm for systems of differential equations. *JOURNAL OF COMPUTATIONAL PHYSICS*, 161(2):401–427, JUL 1 2000. 2.1
- [45] K. Gausling and A. Bartel. Coupling Interfaces and Their Impact in Field/Circuit Co-Simulation. *IEEE Trans. Magn.*, 52(3), MAR 2016. 4.1.2
- [46] K. Gausling and A. Bartel. Density Estimation Techniques in Cosimulation Using Spectral- and Kernel Methods. In Langer, U and Amrhein, W and Zulehner, W, editor, *Scientific Computing in Engineering, SCEE 2016*, volume 28 of *Mathematics in Industry-Cham*, pages 81–89, 2018. 4.1.2
- [47] L. Gerardo-Giorda and F. Nataf. Optimized Schwarz methods for unsymmetric layered problems with strongly discontinuous and anisotropic coefficients. Technical Report 561, CMAP, CNRS UMR 7641, Ecole Polytechnique, France, 2004. submitted. 2.2.1
- [48] D. Gottlieb and C. Shu. On the gibbs phenomenon and its resolution. *SIAM Review*, 39(4):644–668, 1997. 3.2.2.1
- [49] M. Guenther, A. Bartel, B. Jacob, and T. Reis. Dynamic iteration schemes and port-Hamiltonian formulation in coupled differential-algebraic equation circuit simulation. *Int. J. Circuit Theory Appl.*, 49(2):430–452, 2021. 4.1.2
- [50] A. Guironnet, M. Saugier, S. Petitrenaud, F. Xavier, and P. Panciatici. Towards an open-source solution using modelica for time-domain simulation of power systems. In *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pages 1–6. IEEE, 2018. 5.2.1
- [51] R. Hassani, J. Mahseredjian, T. Tshibungu, and U. Karaagac. Evaluation of time-domain and phasor-domain methods for power system transients. *Electric Power Systems Research*, 212:108335, 2022. 1.3

- [52] M.D. Heffernan, K.S. Turner, J. Arrillaga, and C.P. Arnold. Computation of a.c.-d.c. system disturbances - part i and iii, year=1981, volume=PAS-100, number=11, pages=4341-4348, doi=10.1109/TPAS.1981.316825. *IEEE Transactions on Power Apparatus and Systems*. 1.5.1
- [53] A. Ho, C. and Ruehli and P. Brennan. The modified nodal approach to network analysis. *Circuits and Systems, IEEE Transactions on*, 22:504 – 509, 07 1975. 1.3
- [54] K.J. In Hout. On the convergence of wave-form relaxation methods for stiff nonlinear ordinary differential-equations. *Applied Numerical Mathematics*, 18(1-3):175–190, 1995. 4.1.2
- [55] V. Jalili-Marandi, V. Dinavahi, K. Strunz, J. A. Martinez, and A. Ramirez. Interfacing techniques for transient stability and electromagnetic transient programs iee task force on interfacing techniques for simulation tools. *IEEE Transactions on Power Delivery*, 24(4):2385–2395, Oct 2009. 1.4.2, 1.5.1, 1.5.1.1, 1.5.2, 1.5.3.2, 6
- [56] J. Janssen and S. Vandewalle. On SOR waveform relaxation methods. *SIAM Journal on Numerical Analysis*, 34(6):2456–2481, DEC 1997. 4.1.2
- [57] Y.L. Jiang. A general approach to Waveform Relaxation solutions of nonlinear Differential-Algebraic Equations: The continuous-time and discrete-time cases. *IEEE Transaction on Circuits and Systems I*, 51(9):1770–1780, SEP 2004. 4.1.2
- [58] Y.L. Jiang and O. Wing. A note on the spectra and pseudospectra of waveform relaxation operators for linear differential-algebraic equations. *SIAM Journal on Numerical Analysis*, 38(1):186–201, JUN 23 2000. 4.1.2, 4.2, 1, 4.2
- [59] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING*, 48(1):96–129, JAN 10 1998. 1.5.1.3
- [60] H. Konara, U.D. Annakkage, and C. Karawita. Interfacing electromagnetic transient simulation to transient stability model using a multi-port dynamic phasor buffer zone. *CIGRE SCIENCE & ENGINEERING*, page 117, 2019. 1.5.3.1
- [61] MA. Kulasza. *Generalized Dynamic Phasor-Based Simulation for Power Systems*. PhD thesis, Department of Electrical and Computer Engineering University of Manitoba, 2014. 1.3

- [62] T. Ladics. Error analysis of waveform relaxation method for semi-linear partial differential equations. *J. Comput. Appl. Math.*, 285:15–31, SEP 2015. 4.1.2
- [63] P. Le-Huy, G. Sybille, P. Giroux, L. Loud, J. Huang, and I. Kamwa. Real-time electromagnetic transient and transient stability co-simulation based on hybrid line modelling. *IET Generation, Transmission Distribution*, 11(12):2983–2990, 2017. 1.5.1.1, 1.5.3.2
- [64] B. Leimkuhler. Timestep acceleration of waveform relaxation. *SIAM Journal on Numerical Analysis*, 35(1):31–50, FEB 1998. 4.1.2
- [65] E. Lelarasmee, A. Ruehli, and A. Vincentelli. The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1:131–145, 1982. 1.4.1, 4.1.2
- [66] P.-L. Lions. On the Schwarz alternating method. I. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations (Paris, 1987)*, pages 1–42. SIAM, Philadelphia, PA, 1988. 2.2.1, 2.1
- [67] P.-L. Lions. On the Schwarz alternating method. III. A variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations (Houston, TX, 1989)*, pages 202–223. SIAM, Philadelphia, PA, 1990. 2.2.1
- [68] A. Lumsdaine and J.K. White. Accelerating Wave-Form relaxation methods with application to parallel semiconductor-device simulation. *Numer. Funct. Anal. Optim.*, 16(3-4):395–414, 1995. 4.1.2
- [69] A. Lumsdaine and D.Y. Wu. Spectra and pseudospectra of waveform relaxation operators. *SIAM J. Sci. Comput.*, 18(1):286–304, JAN 1997. 4.1.2
- [70] A. Lumsdaine and D.Y. Wu. Krylov subspace acceleration of waveform relaxation. *SIAM Journal on Numerical Analysis*, 41(1):90–111, 2003. 4.1.2
- [71] J. Mahseredjian, S. Denetière, L. Dubé, B. Khodabakhchian, and L. Gérin-Lajoie. On a new approach for the simulation of transients in power systems. *Electric Power Systems Research*, 77(11):1514–1520, 2007. Selected Topics in Power System Transients - Part II. 1.3
- [72] J. Mahseredjian, I. Kocar, and U. Karaagac. *Transient Analysis of Power Systems: Solution Techniques, Tools and Applications*, chapter Solution

- Techniques for Electromagnetic Transients in Power Systems. IEEE Press. Wiley, 2015. 1.2, 1.3
- [73] L. D. Marini and A. Quarteroni. A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.*, 55(5):575–598, 1989. 2.2.1, 2.1
- [74] JR Marti and LR Linares. Real-time emtp-based transients simulation. *IEEE TRANSACTIONS ON POWER SYSTEMS*, 9(3):1309–1317, AUG 1994. IEEE/PES 1993 Summer Meeting, VANCOUVER, CANADA, JUL 18-22, 1993. 2.3.1
- [75] A. Masoom, J. Mahseredjian, T. Ould-Bachir, and A. Guironnet. Electromagnetic transient simulation of large power networks with modelica. 09 2021. 1.2
- [76] S. Mattsson and G. Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM Journal on Scientific Computing*, 14, 05 1993. 5.2.2
- [77] U. Miekkala. Dynamic Iteration methods applied to linear DAE systems. *J. Comput. Appl. Math.*, 25(2):133–151, FEB 1989. 4.1.2, 4.2, 4.2
- [78] U. Miekkala and O. Nevanlinna. Convergence of Dynamic Iteration methods for initial-value problems. *SIAM Journal on Scientific and Statistical Computing*, 8(4):459–482, JUL 1987. 4.1.2
- [79] J. Morales, J. Mahseredjian, A. Ramirez, K. Sheshyekani, and I. Kocar. A loewner/mpm—vf combined rational fitting approach. *IEEE Transactions on Power Delivery*, 35(2):802–808, 2020. 1.5.3.2
- [80] J. Morales Rodriguez, E. Medina, J. Mahseredjian, A. Ramirez, K. Sheshyekani, and I. Kocar. Frequency-domain fitting techniques: A review. *IEEE Transactions on Power Delivery*, 35(3):1102–1110, 2020. 1.5.3.2
- [81] K. Mudiyansele Harshani. *Interfacing a Transient Stability Model to a Real-Time Electromagnetic Transient Simulation Using Dynamic Phasors*. PhD thesis, Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, 2019. 3.2.1.1
- [82] K. Mudunkotuwa. *Co-simulation of power system transients using dynamic phasor and electromagnetic transient simulators*. PhD thesis, Department of Electrical and Computer Engineering, Faculty of Engineering, University of Manitoba, Winnipeg, 2018. 1.5.1.1, 3.2.1.1

- [83] K. Mudunkotuwa, S. Filizadeh, and U. Annakkage. Development of a hybrid simulator by interfacing dynamic phasors with electromagnetic transient simulation. *IET Generation, Transmission & Distribution*, 11, 03 2017. 1.5.3.1
- [84] F. Nataf. Quasi Optimal Interface Conditions in Domain Decomposition Methods. Application to Problems with Extreme Contrasts in the Coefficients. R.I. 514, CMAP, Ecole Polytechnique, October 2003. 2.2.1
- [85] F. Nataf, H. Xiang, V. Dolean, and N. Spillane. A coarse space construction based on local Dirichlet-to-Neumann maps. *SIAM J. Sci. Comput.*, 33(4):1623–1642, 2011. 2.2.1
- [86] R. Natarajan. Domain decomposition using spectral expansions of Steklov-Poincaré operators. II. A matrix formulation. *SIAM J. Sci. Comput.*, 18(4):1187–1199, 1997. 2.2.2
- [87] M. Netto, Y. Susuki, and L. Mili. Data-driven participation factors for nonlinear systems based on koopman mode decomposition. *IEEE Control Systems Letters*, 3(1):198–203, 2019. 1.5.1.2
- [88] A. Nicolai. Co-simulation-test case: Predator-prey (lotka-volterra) system. Technical report, Institut fur Bauklimatik, Technische Universität Dresden, 2019. 5.4.2.2
- [89] A. Nicolai, A. Paepcke, and H. Hirsch. Robust and accurate co-simulation master algorithms applied to fmi slaves with discontinuous signals using fmi 2.0 features. pages 769–776, 02 2019. 1.4.1
- [90] University of Minnesota. Parmetis, 1997. 1.5.1.3
- [91] J. Pade and C. Tischendorf. Waveform relaxation: a convergence criterion for differential-algebraic equations. *Numer. Algorithms*, 81(4, SI):1327–1342, 2019. 3.4.1.1, 4.1.2, 4.5
- [92] V. Paduani, B. Xu, D. Lubkeman, and N. Lu. Novel Real-Time EMT-TS Modeling Architecture for Feeder Blackstart Simulations, 2021. 1.5.3.1
- [93] E. Perez-Inigo Ramirez, A. Benchaib, and J. Gonzalez. Modelling and dynamic equivalents of an ac network. Conference presentation. 1.1, 6
- [94] F. Plumier. *Co-simulation of Electromagnetic Transients and Phasor Models of Electric Power Systems*. PhD thesis, ULiège - Université de Liège, 29 January 2016. 1.5.3.1, 3.2.1.1

- [95] F. Plumier, P. Aristidou, C. Geuzaine, and T. Van Cutsem. Co-Simulation of Electromagnetic Transients and Phasor Models: A Relaxation Approach. *IEEE Trans. on Power Delivery*, 31(5):2360–2369, 2016. 1.5.2, 1.5.3.2
- [96] F. Plumier, P. Aristidou, Ch. Geuzaine, and Th. Van Cutsem. A relaxation scheme to combine phasor-mode and electromagnetic transients simulations. In *Proceedings - 2014 Power Systems Computation Conference, PSCC 2014*, 08 2014. 1.5.3.1
- [97] B. Porkar, M. Vakilian, and R. Feuillet. Frequency-dependent network equivalent for electromagnetic transient studies by vector fitting. pages 166 – 171, 06 2006. 1.5.3.2
- [98] F. Pruvost, P. Laurent-Gengoux, F. Magoulés, and B. Haut. Accelerated Waveform Relaxation methods for power systems. In *2011 International Conference on Electrical and Control Engineering*, pages 2877–2880, 2011. 1.4.1
- [99] A Quarteroni. Domain decomposition methods. In *NUMERICAL MODELS FOR DIFFERENTIAL PROBLEMS*, volume 2 of *MS&A-Modeling Simulation and Applications*, pages 501–546. 2009. 2.2.1
- [100] F. Raak, Y. Susuki, T. Hikihara, H. R. Chamorro, and M. Ghandhari. Partitioning power grids via nonlinear koopman mode analysis. In *ISGT 2014*, pages 1–5, 2014. 1.5.1.2
- [101] M.W. Reichelt, J.K. White, and J. Allen. Optimal convolution SOR acceleration of Wave-Form relaxation with application to parallelsimulation of semiconductor-devices . *SIAM J. Sci. Comput.*, 16(5):1137–1158, SEP 1995. 4.1.2
- [102] J. Rimorov, D. and Huang, C.F. Mugombozi, T. Roudier, and I. Kamwa. Power coupling for transient stability and electromagnetic transient collaborative simulation of power grids. *IEEE Transactions on Power Systems*, 36(6):5175–5184, 2021. 1.5.3, 5.4.3
- [103] AR Sana, J Mahseredjian, X Daido, and H Dommel. Treatment of discontinuities in time-dimain simulation of switched networks. *Mathematics and Computers in simulation*, 38(4-6):377–387, AUG 1995. 2.3.1
- [104] S. Schoeps, H. De Gersem, and A. Bartel. Higher-Order Cosimulation of Field/Circuit Coupled Problems. *IEEE Transactions in Magnetics*, 48(2):535–538, 2012. 4.1.2

- [105] H. A. Schwarz. Gesammelte Mathematische Abhandlungen. *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, 15:272–286, 1870. 2.1, 2.2.1
- [106] D. Shu, X. Xie, Q. Jiang, Q. Huang, and C. Zhang. A novel interfacing technique for distributed hybrid simulations combining emt and transient stability models. *IEEE Transactions on Power Delivery*, 33(1):130–140, Feb 2018. 1.5.2
- [107] J. Shu, Wei Xue, and Weimin Zheng. A parallel transient stability simulation for power systems. *IEEE Transactions on Power Systems*, 20(4):1709–1717, 2005. 1.5.1.3
- [108] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl. Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps. *Numerische Mathematik*, 126(4):741–770, APR 2014. 2.2.1
- [109] A. St-Cyr, M. J. Gander, and S. J. Thomas. Optimized multiplicative, additive, and restricted additive Schwarz preconditioning. *SIAM J. Sci. Comput.*, 29(6):2402–2425 (electronic), 2007. 2.2.1, 2.1
- [110] D. Tromeur-Dervout. Meshfree Adaptative Aitken-Schwarz Domain Decomposition with application to Darcy Flow. In Topping, BHV and Ivanyi, P, editor, *Parallel, Distributed and Grid Computing for Engineering*, volume 21 of *CSET Series*, pages 217–250. Saxe-Coburg Publications, 2009. 3.3.1
- [111] D. Tromeur-Dervout. Approximating the trace of iterative solutions at the interfaces with nonuniform Fourier transform and singular value decomposition for cost-effectively accelerating the convergence of Schwarz domain decomposition. *ESAIM: Proc.*, 42:34–60, 2013. 3.3.1
- [112] T. Tsuji, F. Magoulès, K. Uchida, and T. Oyama. A Partitioning Technique for a Waveform Relaxation Method Using Eigenvectors in the Transient Stability Analysis of Power Systems. *IEEE Transactions on Power Systems*, 30(6):2867–2879, 2015. 1.5.1.2
- [113] A. A. van der Meer, M. Gibescu, M. A. M. M. van der Meijden, W. L. Kling, and J. A. Ferreira. Advanced hybrid transient stability and emt simulation for vsc-hvdc systems. *IEEE Transactions on Power Delivery*, 30(3):1057–1066, June 2015. 1.5.2

- [114] L. Wedepohl and L. Jackson. Modified nodal analysis: an essential addition to electrical circuit theory and analysis. *Engineering Science and Education Journal*, 11(3):84–92, 2002. 1.3
- [115] M. Zamroni. *Development of EMT/TS Co-simulation Using PowerFactory and PSS/E*. PhD thesis, Delft University of Technology, 2017. 3.2.1.1
- [116] F. Znidi, H. Davarikia, M. Arani, and M. Barati. Coherency Detection and Network Partitioning based on Hierarchical DBSCAN. In *2020 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–5, 2020. 1.5.1.2

List of Figures

1.1	Time frame of power system transients [93]	4
2.1	Linear RLC circuit and its associated EMT modeling DAE system with $x = \{v_1, i_{23}, v_4, v_5, i_{67}, v_7\}$ and $y = \{v_2, i_{12}, v_3, i_{34}, i_{45}, i_{56}, v_6, i_{71}\}$. $L1 = L2 = 0.7$, $C1 = C2 = 1.10^{-6}$, $R1 = R2 = 77$, $Z_s = 1.10^{-6}$, $\omega_0 = 2\pi 50$	37
2.2	Graph partitioning of the RLC circuit in two subdomains and the associated matrix partitioning without overlap (top) and with overlap of 1 (bottom). EMT case	38
2.3	Error between two consecutive RAS iterations for EMT-EMT case with top $\Delta t_{emt} = 10^{-4}$, bottom $\Delta t_{emt} = 10^{-4}$. By taking all the errors for the two on the right and taking only one out of two for the two on the left.	40
2.4	Error between two consecutive RAS iterations for TS-TS case with top $\Delta t_{emt} = 10^{-4}$, bottom $\Delta t_{emt} = 10^{-4}$. By taking all the errors for the two on the right and taking only one out of two for the two on the left.	41
2.5	Homogeneous DDM results comparison with DAE monodomain: (Top) RAS for EMT modeling with $\Delta t_{emt} = 1.10^{-4}$ and (Bottom) RAS for TS modeling with $\Delta t_{ts} = 2.10^{-4}$	43
2.6	Homogeneous DDM results comparison with DAE monodomain: (Top) RAS for EMT modeling with $\Delta t_{emt} = 1.10^{-2}$ and (Bottom) RAS for TS modeling with $\Delta t_{ts} = 2.10^{-2}$	44
3.1	1 st Harmonic of a moving history, computed with PhasorDynamic and an FFT. Top: without applying any correction, Bottom: with applying a correction	52
3.2	Impact of an EMT event in the History	54

- 3.3 TS-EMT hybrid simulation of the test of section 2.5 with a jump in the voltage source at $t = 0.2s$ that last less than Δt_{TS} : without (top) and with (bottom) linear interpolation of \mathbb{W}_{TS}^{N+1} 55
- 3.4 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with $C_1 = C_2 = 1.10^{-6}$, $R_1 = R_2 = 7, L_1 = 0.07$ and we have $L_2 = 1$ at the top and $L_2 = 0.07$ at the bottom. 62
- 3.5 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.4$ and we have $C_1 = C_2 = 1.10^{-6}$ at the top and $C_1 = C_2 = 1.10^{-4}$ at the bottom. 63
- 3.6 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$. At the top the source is in the EMT subsystem and at the bottom the source is in the TS subsystem. 64
- 3.7 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the TS subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$ at the top $\Delta t_{emt} = 2.10^{-3}$ and at the bottom $\Delta t_{emt} = 2.10^{-5}$ 65
- 3.8 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with voltage source in the EMT subsystem and with, $R_1 = R_2 = 7, L_1 = L_2 = 0.5, C_1 = C_2 = 1.10^{-6}$ at the top $\Delta t_{emt} = 2.10^{-3}$ and at the bottom $\Delta t_{emt} = 2.10^{-5}$ 66
- 3.9 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the α parameter in the smoothing and with voltage source in the EMT subsystem. $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$. With top left $\alpha = 0\%$, top right $\alpha = 25\%$, bottom left $\alpha = 50\%$ and bottom right $\alpha = 75\%$. . . 67
- 3.10 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the α parameter in the smoothing and with voltage source in the TS subsystem. With, $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$. With top left $\alpha = 0$, top right $\alpha = 25$, bottom left $\alpha = 50$ and bottom right $\alpha = 75$ 68
- 3.11 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the $\Delta t_{ts} = 10^{-2}$ less than a the period T and rolling history, with the source voltage in EMT and $\alpha = 0\%$ (top), $\alpha = 25\%$ (bottom), $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$ 69

3.12 Convergence of the heterogeneous EMT-TS RAS error between two consecutive iterations with respect to the $\Delta t_{ts} = \Delta t_{ts} = 10^{-2}$ less than a the period T and rolling history, with the source voltage in TS and $\alpha = 0\%$ (top), $\alpha = 25\%$ (bottom), $R_1 = R_2 = 7, L_1 = 0.3, L_2 = 0.7, C_1 = C_2 = 1.10^{-6}$ 70

3.13 Heterogeneous EMT-TS RAS convergence error for for each subdomain, for the time step $t = 0.02$ and its Aitken's acceleration applied on the TS partition interface with $\Delta t_{ts} = 2.10^{-2}$ and $\Delta t_{emt} = 2.10^{-4}$) and with parameters $L_1 = 0.07, C1 = 1.10^{-6}, R1 = 7, L2 = 0.07, C2 = 1.10^{-6}, R2 = 7, Z_s = 0.000001$, with the voltage source in the TS part 72

3.14 Heterogeneous EMT-TS RAS convergence error for the TS boundary for the time step $t = 0.02$ and its Aitken's acceleration applied to the TS partition interface with $\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$) and with parameters $L_1 = 0.07, C1 = 1.10^{-6}, R1 = 7, L2 = 0.07, C2 = 1.10^{-6}, R2 = 7, Z_s = 0.000001$, with the voltage source in the EMT part 73

3.15 Heterogeneous EMT-TS RAS solution comparison with the monolithic EMT case on a time intervalle with $\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$) and with parameters $L_1 = 0.07, C1 = 1.10^{-6}, R1 = 7, L2 = 0.07, C2 = 1.10^{-6}, R2 = 7, Z_s = 0.000001$, with the voltage source in the EMT part 74

3.16 Comparison of the behavior, with respect to time, of the variables i_{34} (top left) and v_3 (bottom left) (the figures on the right are their zoom on the disturbances) computed using the heterogeneous EMT-TS RAS splitting with the Aitken's technique for accelerating convergence ($\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$), the reference is the monolithic EMT. An amplitude perturbation on the voltage source starting at $t = 0.02s$ and ending at $t = 0.021s$, therefore lasting less than one Δt_{ts} is applied. Parameters are $L_1 = 0.07, C_1 = 1.10^{-5}, R_1 = 7, L_2 = 0.07, C_2 = 1.10^{-7}, R_2 = 7, Z_s = 0.000001$ 75

3.17 Comparison of the behavior, with respect to time, of the variables v_3 (left) and v_7 (right) computed using the heterogeneous EMT-TS RAS splitting with the Aitken's technique for accelerating convergence ($\Delta t_{ts} = 2.10^{-3}$ and $\Delta t_{emt} = 2.10^{-5}$), the reference is the monolithic EMT. An amplitude perturbation on the voltage source. Parameters are $L_1 = 0.07, C_1 = 1.10^{-5}, R_1 = 7, L_2 = 0.07, C_2 = 1.10^{-7}, R_2 = 7, Z_s = 0.000001$ 76

4.1	(top) Sequential DI with the RAS splitting convergence behavior with and without Aitken's acceleration ($\log_{10}(\ z^{(2k)} - z_{ref}\ _{\infty})$) with respect to the iterations for different values of Δt and one time step. (bottom) Comparison between the sequential DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference for the e_3 variable with $\Delta t = 1.2 \cdot 10^{-3}$, with parameters: $L_1 = 0.4$, $L_2 = 0.5$, $C = 1. \cdot 10^{-6}$, $G = 2. \cdot 10^{-3}$.	99
4.2	Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference for the i_1 variable computed with a sequential strategy with a non linear component, with $L_1 = 7$, $L_2 = 0.7$, $C = 1. \cdot 10^{-6}$, $G_0 = 0.07$, $\alpha = 605$	100
4.3	Evolution of the spectral radius of the error operator depending on the number of pipelined regular time steps of size $\Delta t = 1.1 \cdot 10^{-3}$ (left, $\Xi = 14$) and $\Delta t = 1.1 \cdot 10^{-4}$ (right, $\Xi = 120$), with $L_1 = 0.4$, $L_2 = 0.7$, $C = 1. \cdot 10^{-6}$, $G = 2. \cdot 10^{-3}$	101
4.4	(Left) DI with the RAS splitting convergence behavior ($\log_{10}(\ z^{(2k)} - z_{ref}\ _{\infty})$) on each of the pipelined time steps with respect to the iterates and (right) comparison between the DI with the RAS splitting with and without the Aitken's technique for accelerating convergence and the DAE monolithic reference for the e_3 variable with $\Delta t = 1, 1. \cdot 10^{-3}$, $\Xi = 14$, with parameters : $L_1 = 0.4$, $L_2 = 0.5$, $C = 1. \cdot 10^{-6}$, $G = 2. \cdot 10^{-3}$	102
4.5	Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference with a non-linear component, with $\Delta t = 1, 1. \cdot 10^{-4}$, $\Xi = 100$, with parameters: $L_1 = 2.7$, $L_2 = 0.9$, $C = 1. \cdot 10^{-6}$, $G_0 = 0.07$, $\alpha = 605$	104
4.6	Comparison between the DI with the RAS splitting with the Aitken's technique for accelerating convergence and the DAE monolithic reference with $\Delta t = 1, 1. \cdot 10^{-3}$ from $t = 0$ to $t = 0.008$, $\Delta t = 1, 1. \cdot 10^{-4}$ from $t = t = 0.0081$ to $t = 0.0101$ then $\Delta t = 1, 1. \cdot 10^{-3}$ from $t = 0.0111$ to the end, $\Xi = 48$, with parameters: $L_1 = 5$, $L_2 = 0.7$, $C = 1. \cdot 10^{-6}$, $G = 1. \cdot 10^{-6}$	105
5.1	The Modelica Library	110
5.2	Different representations of the same class "circuit" under OpenModelica	111
5.3	Instantiation of the class "PetitCircuit" on OpenModelica , a part has been cut, the complete instantiation is composed of 131 lines	112

5.4	The modelica Library	115
5.5	The modelica Library	117
5.6	Three Buses monoblock model chosen to test the interface component under OMEdit	118
5.7	Three Buses model with interface components under OMEdit	119
5.8	Test on the causality of values	121
5.9	Class Diagram of a model to be exported	122
5.10	Modeldescription variable with the same reference value	123
5.11	Co-simulation platform architecture with MPI Master-Slaves communications and using local solver in the FMI standard or C++ DAE functions.	126
5.12	Class Diagram of the Architecture	128
5.13	Detailed Class Diagram of the Architecture.	129
5.14	FMI 2 functions	131
5.15	FMI 2 functions for Cosimulation	132
5.16	FMI 2 functions for Model Exchange	132
5.17	Pseudo Code representing the Co simulation of two FMU slave	133
5.18	C++ wrapper validation results using the Lokta-Voltera with $\alpha = \frac{2}{3}, \beta = \frac{4}{3}, \delta = 1, \gamma = 1$	135

Introduction de la problématique et résumé de la thèse

Actuellement, l'écologie est au cœur des préoccupations, l'intégration des énergies renouvelables dans le réseau se développe. L'interconnexion entre des zones géographique de plus en plus éloignées se développe. Pour ce faire, des lignes Haute Tension Courant Continu (HVDC) sont créées. Or, les lignes HVDC sont connectées aux réseaux via des convertisseurs électroniques de puissance.

Ce développement des énergies renouvelables et des liaisons HVDC conduit donc une présence grandissante de composants d'électronique de puissance dans les réseaux. Par ailleurs, ce type de technologie rend plus courant dans le réseau des phénomènes impliquant des transitoires plus rapides que les transitoires mises en jeu jusqu'alors. Cette évolution du réseau doit s'accompagner d'une évolution des méthodes utilisées pour le simuler.

Dans cette perspective, la co-simulation EMT-TS est envisagée comme une des solutions pour adapter les simulations aux réseaux actuels. En effet, les simulations de type électromagnétiques transitoires (EMT) peuvent simuler un vaste éventail de fréquence avec des événements de l'ordre de la micro-seconde. Cependant, les pas de temps effectués sont forcément plus petits que la durée des événements à simuler et sont donc très petits ce qui rend la simulation très coûteuse en temps de calcul. Les simulations de type stabilité transitoire sont le plus souvent réalisées en utilisant des phaseurs, il y a donc une hypothèse forte qui est faite sur la forme des solutions, la plupart des oscillations sont lissées. Cette représentation est suffisante pour les événements allant jusqu'à la milli-seconde elle n'est donc pas suffisamment précise pour simuler les transitoires induites par les composants d'électronique de puissance. En revanche, ce type de simulation à l'avantage d'être peu coûteuse en temps de calcul et demande peu de ressources.

L'idée de la co-simulation EMT-TS est donc de découper le réseau et de simuler une petite partie choisie comme étant un domaine d'intérêt en utilisant la simulation EMT et le reste du réseau en utilisant une simulation de type TS, et ainsi de tirer parti à la fois du niveau de détail de la simulation de type EMT et à la fois de l'efficacité de calcul du TS.

Pour mettre en place une co-simulation EMT-TS, plusieurs verrous doivent être levés, Jalili-Marandi & al [55] les exposent :

- Le partitionnement : avec un bon compromis détail/efficacité, il faut donc limiter autant que possible la taille du domaine EMT. Le découpage doit être suffisamment éloigné des perturbations potentielles à la fois pour capter les perturbations éventuelles avec la modélisation EMT et pour la stabilité de la méthode, en effet si l'interface est trop proche de la source de la perturbation, cela peut poser des problèmes dans la transmission des données. La décomposition doit également permettre à la méthode de converger, effectivement beaucoup de partitionnements sont choisis de sorte à rendre l'opérateur d'erreur de la méthode de Co-simulation contractant.
- Le protocole d'interaction : une difficulté particulière à la co-simulation EMT-TS dans le choix du protocole est le fait que les simulations EMT et les simulations TS n'utilisent pas du tout les mêmes pas de temps. Dans la plupart des co-simulations EMT-TS, les pas de temps sont choisis de sorte à ce que $\Delta t_{ts} = n\Delta t_{emt}$ avec $n \in \mathbb{N}^*$. Le macro-pas de temps de la méthode est souvent choisi comme étant le pas de temps TS Δt_{ts} .

Plus généralement, le protocole peut être en série ou en parallèle, les mises à jour des sous-modèles avec les données récupérées de l'autre sous-modèle peuvent être synchronisées ou non et plus ou moins fréquentes. Le protocole peut être itératif ou non.

- La traduction d'un type de modèle à un autre : en effet, lors de la simulation EMT, on ne fait pas d'hypothèse sur la forme des variables, alors que dans le cas TS, on suppose que les variables sont de forme phaseur. Par conséquent, les variables doivent être transformées pour l'échange d'informations afin d'être compréhensibles par chacun des sous-domaines. Ces transformations ne doivent pas être trop coûteuses en calcul et aussi précises que possible.

Dans cette thèse, l'accent est mis sur l'utilisation de méthode de décomposition de domaine au service de ce sujet de co-simulation hétérogène, la thèse est écrite comme suit :

Dans un premier temps, le sujet et la problématique sont introduits. Un état de l'art concernant les différents verrous cités plus haut est exposé. Une méthode pour obtenir le système d'équations algébrique différentiel représentant le réseau est introduite.

Ensuite, les méthodes de décomposition de domaine de type Schwarz qui seront utilisées tout au long de la thèse sont résumés. L'application de ce type de méthode sur les circuits, dans des cas homogène est détaillée. La convergence de ces méthodes est étudiée et est purement linéaire. Cette propriété nous permet d'utiliser la

méthode d'accélération de la convergence d'Aitken pour accélérer la convergence vers la vraie solution et ceci que la méthode converge ou diverge.

Puis, la méthode utilisée est adaptée au cas hétérogène. Ainsi, le protocole est modifié pour répondre à la différence de pas de temps utilisé par chacun des sous-domaines. Des opérateurs de traductions entre l'EMT et le TS sont choisis et améliorés pour répondre à nos exigences. On voit alors que tant que ces opérateurs sont linéaires la convergence/divergence reste linéaire et la technique d'accélération de la convergence d'Aitken peut être appliquée. Une stratégie adaptée aux contraintes liée à l'hétérogénéité EMT-TS est mise en place afin d'utiliser la technique d'accélération d'Aitken de façon optimale.

Dans un quatrième temps, la méthode de Schwarz est remplacée dans le contexte plus général de la dynamique itération avec un découpage particulier. Il est alors montré que la technique d'accélération de la convergence d'Aitken peut être utilisée pour accélérer plusieurs pas de temps à la fois. Ce protocole se révèle particulièrement intéressant dans le cas où il y aurait des composants non-linéaires dans le circuit. Enfin, dans un dernier chapitre, un retour d'utilisation de la suite d'outils OpenModelica est présenté. Une plateforme en cours de développement désigné dans une architecture master/slave est présentée. Le cœur du master est basé sur les méthodes développées dans cette thèse.

Abstract

The integration of renewable energies in the electrical network leads to an increasing presence of power electronic components, which makes phenomena involving electromagnetic transients (EMT), faster than electromechanical transients, more frequent than until recently. This evolution of the network must be accompanied by an evolution of the simulation methods. This thesis focuses on the development of numerical methods for EMT-TS co-simulation, taking advantage of the detailed but time-consuming EMT simulation and the computationally efficient TS simulation, which captures only smooth oscillations, to simulate different parts of the network. Several numerical difficulties appear in such a coupling to solve systems of differential algebraic equations (DAE): the partitioning can have an impact on the contraction of the error operator of the numerical method, the difference in solver time step between EMT-TS simulations requires an interaction protocol to avoid any delay in data exchange, the difference in mathematical modeling requires a specific translation of the data to be exchanged. We develop a restrictive additive Schwarz domain decomposition method to simulate linear RLC circuits or their linearization around each time step if there are non-linear components. Convergence or divergence studies in the EMT-EMT or TS-TS homogeneous modeling cases show that they depend on the partitioning, the time step size, the values of the components circuits. Nevertheless, we take advantage of their purely linear behavior in order to apply the Aitken convergence acceleration technique and thus obtain simulation results in a limited number of iterations even in the case of a divergent method. The constraint imposed on the fractionation to have a convergent method is then no longer relevant. Heterogeneous EMT-TS splitting requires that the RAS take into account the difference in time step between the two simulations. We develop translation operators, as accurate as possible, not too expensive, without additional constraints on the time steps used, and finally linear in order to keep the linear convergence/divergence and thus the advantage of being able to use the Aitken convergence acceleration technique. Then, we show that our domain decomposition method applied to the DAE system is a special case of a general method called dynamic iteration. We then show that dynamic iteration with the domain decomposition method can be accelerated using Aitken's convergence acceleration technique and in particular that several time steps can be accelerated at the same time even if they have variable sizes programmed in advance, and in the presence of nonlinear components in the circuit. Finally, we study the advantages and disadvantages of Modelica-based tools such as OpenModelica to implement our domain decomposition. These tools allow us to produce the DAE system from the mathematical description of the network components. The result is the development of a platform using a master/slave implementation of our domain decomposition method, through the MPI (Message Passing Interface) library. This platform is intended to be as general as possible, using either models (part of the network) in the form of a functional mockup unit (FMU) from industrial modeling tools, or a DAE system coded in C++.

